# 单机优化之随机算法

焦思逸 2020.7.8

1. 基本随机优化算法
- 随机梯度下降
- 随机坐标下降
- 随机拟牛顿法
- 对偶坐标上升法

2. 随机优化算法的改进
- 方差缩减
- 算法组合

3. 非凸随机优化算法
- Ada系列算法
- 非凸理论分析
- 逃离鞍点问题
- 等级优化算法

# 1. 基本随机优化算法：SGD

**算法 5.1　随机梯度下降法**

**Initialize**：$w_0$

**Iterate**：for $t = 0, 1, \cdots, T-1$

1. 随机选取一个样本 $i_t \in \{1, \cdots, n\}$
2. 计算梯度 $\nabla f_{i_t}(w_t)$
3. 更新参数 $w_{t+1} = w_t - \eta_t \nabla f_{i_t}(w_t)$

**end**

$$\mathbb{E}_{i_t} \nabla f_{i_t}(w_t) = \nabla f(w_t)$$

定理5.1　假设目标函数$f$是$R^d$上的凸函数，并且$L$-Lipschitz连续，$w^* = \arg\min_{\|w\| \leqslant D} f(w)$，当步长$\eta_t = \sqrt{\frac{D^2}{L^2 t}}$时，随机梯度下降法具有如下次线性收敛速率：

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} f(w_t) - f(w^*)\right] \leqslant \frac{LD}{\sqrt{T}}$$

**定理5.2** 假设目标函数$f$是$R^d$上的$\alpha$-强凸函数，并且$\beta$-光滑，如果随机梯度的二阶矩有上界，即$\mathbb{E}_{i_t}\|\nabla f_{i_t}(w_t)\|^2 \leqslant G^2$，当步长$\eta_t = \frac{1}{\alpha t}$时，随机梯度下降法具有如下线性收敛速率：

$$\mathbb{E}[f(w_T) - f(w^*)] \leqslant \frac{2\beta G^2}{\alpha^2 T}$$

**算法 5.2　小批量随机梯度下降法**

**Initialize**：$w_0$

**Iterate**：for $t=0, 1, \cdots, T-1$

1. 随机选取一个小批量样本集合 $S_t \subset \{1, \cdots, n\}$
2. 计算梯度 $\nabla f_{S_t}(w_t) = \frac{1}{|S_t|}\sum_{i \in S_t} \nabla f_i(w_t)$
3. 更新参数 $w_{t+1} = w_t - \eta_t \nabla f_{S_t}(w_t)$

**end**

# 1. 基本随机优化算法：随机坐标下降法

**算法 5.3　随机坐标下降法**

**Initialize**：$w_0$

**Iterate**：for $t=0, 1, \cdots, T-1$

1. 随机选取一个维度 $j_t \in \{1, \cdots, d\}$
2. 计算梯度 $\nabla_{j_t} f(w_t)$
3. 更新参数：

$$w_{t+1,j_t} = w_{t,j_t} - \eta_t \nabla_{j_t} f(w_t)$$

**end**

$$\mathbb{E}_{j_t} \nabla_{j_t} f(w_t) = \frac{1}{d} \nabla f(w_t)$$

定义 5.1　如果对任意模型 $w \in R^d$，对于维度 $j$ 存在常数 $\beta_j$，使得 $\forall \delta \in R$ 有下面不等式成立：

$$|\nabla_j f(w + \delta e_j) - \nabla_j f(w)| \leqslant \beta_j |\delta|$$

则称目标函数 $f$ 对于维度 $j$ 具有 $\beta_j$-Lipschitz 连续的偏导数。

如果 $f$ 对于每个维度的偏导数都是 Lipschitz 连续的，我们记 $\beta_{\max} = \max_{j=1,\cdots,d} \beta_j$。

定理5.3　假设目标函数$f$是$R^d$上的凸函数，并且具有$\beta_j$-Lipschitz连续的偏导数，记$w^* = \arg\min_{||w|| \leqslant D} f(w)$，当步长$\eta = 1/\beta_{\max}$时，随机坐标下降法具有如下的次线性收敛速率：

$$\mathbb{E} f(w_T) - f(w^*) \leqslant \frac{2d\beta_{\max} D^2}{T}$$

定理 5.4　假设目标函数 $f$ 是 $R^d$ 上的 $\alpha$-强凸函数，并且具有 $\beta_j$-Lipschitz 连续的偏导数，当步长 $\eta = 1/\beta_{\max}$ 时，随机坐标下降法具有如下的线性收敛速率：

$$\mathbb{E} f(w_T) - f(w^*) \leqslant \left(1 - \frac{\alpha}{d\beta_{\max}}\right)^T (f(w_0) - f(w^*))$$

**算法 5.4　随机块坐标下降法**

**Initialize**：$w_0$

将 $d$ 个维度均等切分为 $J$ 块

**Iterate**：for $t=0, 1, \cdots, T-1$

1. 随机选取一块 $J_t \in \{1, \cdots, J\}$
2. 计算梯度 $\nabla_{J_t} f(w_t)$
3. 更新参数 $w_{t+1,j} = w_{t,j} - \eta_t \nabla_{J_t} f(w_t), j \in J_t$

**end**

# 1. 基本随机优化算法：随机拟牛顿法

$$w_{t+1} = w_t - \eta_t H_t \left( \frac{1}{b} \sum_{i \in S_t} \nabla f_i(w_t) \right)$$

定理 5.5　假设上述条件（1）~（4）成立，当步长 $\eta_t = a/t$ 并且 $a > \frac{1}{2\mu_1\lambda_1}$ 时，随机拟牛顿法具有如下次线性收敛速率：

$$\mathbb{E}f(w_T) - f(w^*) \leqslant \frac{Q(a)}{T}$$

其中 $Q(a) = \max\left\{ \frac{\lambda_2 \mu_2^2 a^2 G^2}{2(2\mu_1\lambda_1 a - 1)}, f(w_1) - f(w^*) \right\}$。

Byrd R H, Hansen S L, Nocedal J, et al. A stochastic quasi-Newton method for large-scale optimization[J]. SIAM Journal on Optimization, 2016, 26(2): 1008-1031.

# 1. 基本随机优化算法：随机对偶坐标上升法

**算法 5.7　随机对偶坐标上升法**

**Initialize**：$\alpha_0$，$w_0 = w(\alpha_0)$

**Iterate**：for $t = 0, 1, \cdots, T-1$

1. 随机抽取一个样本 $i_t \in \{1, \cdots, n\}$
2. 求解子问题，找到

$$\Delta \alpha_{i_t} = \arg\max_z \left\{ -\phi_{i_t}^*(-\alpha_{t,i} + z) - \frac{\lambda n}{2} \left\| w_t + \frac{1}{\lambda n} z x_{i_t} \right\|^2 \right\}$$

3. 更新参数 $\alpha_{t+1} = \alpha_t + \Delta \alpha_{i_t} e_{i_t}$，$w_{t+1} = w_t + \frac{1}{\lambda n} \Delta \alpha_{i_t} x_{i_t}$

**Output(i)**：$\bar{\alpha} = \frac{1}{T - T_0} \sum_{t=T_0+1}^{T} \alpha_t$；$\bar{w} = w(\bar{\alpha}) = \frac{1}{T - T_0} \sum_{t=T_0+1}^{T} \alpha_t x_{i_t}$

**Output(ii)**：随机选取一个 $t \in \{T_0 + 1, \cdots, T\}$，$\bar{\alpha} = \alpha_t$，$\bar{w} = w_t$

**end**

Shalev-Shwartz S, Zhang T. Stochastic dual coordinate ascent methods for regularized loss minimization[J]. Journal of Machine Learning Research, 2013, 14(Feb): 567-599.

# 2. 随机优化算法的改进：方差缩减方法

$$\text{SVRG}: \quad w_{t+1} = w_t - \eta\left(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}) + \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{w})\right)$$

**算法 5.8　SVRG 算法**

**Initialize**：$\tilde{w}_0$

**Iterate**：for $s = 0, 1, 2, \cdots, S-1$

1. $\tilde{w} = \tilde{w}_{s-1}$
2. 计算准确的梯度：$\tilde{u} = \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\tilde{w})$
3. $w_0 = \tilde{w}$

**Iterate**：for $t = 0, 1, \cdots, M-1$

4. 随机选取一个样本 $i_t \in \{1, \cdots, n\}$
5. 更新参数 $w_{t+1} = w_t - \eta(\nabla f_{i_t}(w_t) - \nabla f_{i_t}(\tilde{w}) + \tilde{u})$

**end**

**Output**（**i**）：$\tilde{w}_S = w_M$

**Output**（**ii**）：随机选取一个 $t \in \{1, \cdots, M\}$，$\tilde{w}_S = w_t$

**end**

**定理 5.7** 如果目标函数是 $\alpha$-强凸并且 $\beta$-光滑的，当步长 $\eta \leqslant \frac{1}{\beta}$ 时，SVRG 算法的收敛速率为：

$$\mathbb{E}[f(\widetilde{w}_s) - f(w^*)] \leqslant \left(\frac{1}{\alpha\eta(1-2\beta\eta)M} + \frac{2\beta\eta}{1-2\beta\eta}\right)^s \mathbb{E}[f(\widetilde{w}_0) - f(w^*)]$$

其中 $M$ 为内循环轮数。

# 2. 随机优化算法的改进：方差缩减方法

（1）小批量采样方法

小批量采样方法每次抽取多个样本（多于单样本，少于全样本），在随机优化算法和确定性优化算法之间寻找某种折中。相比于确定性优化算法，小批量随机算法可以提高更新的速度，减小运算复杂度；相比于随机算法，小批量随机算法因为使用多个样本来计算梯度，可以降低随机梯度的方差，提高收敛速率。

（2）带权重的采样方法

除了均匀的有放回采样以外，适当地改变采样的权重可以有效地提高收敛速率。Peilin Zhao 和 Tong Zhang 提出了基于重要性采样的随机梯度下降法[11]。

# 2. 随机优化算法的改进：算法组合方法

1. 与SVRG组合
- SVRF : Hazan E, Luo H. Variance-reduced and projection-free stochastic optimization[C]//International Conference on Machine Learning. 2016: 1263-1271.
- SVRG-ADMM : Zheng, Shuai, and James T. Kwok. "Fast-and-Light Stochastic ADMM." IJCAI. 2016.
- 方差缩减的BFGS : Gower, Robert, Donald Goldfarb, and Peter Richtárik. "Stochastic block BFGS: Squeezing more curvature out of data." *International Conference on Machine Learning*. 2016.

2. 与Nesterov加速法组合
- APCG : Lin, Qihang, Zhaosong Lu, and Lin Xiao. "An accelerated proximal coordinate gradient method." *Advances in Neural Information Processing Systems*. 2014.
- 小批量加速近端随机方差缩减梯度法 : Nitanda, Atsushi. "Stochastic proximal gradient descent with acceleration techniques." *Advances in Neural Information Processing Systems*. 2014.

3. 与随机坐标下降法组合
- MRBCD : Zhao, Tuo, et al. "Accelerated mini-batch randomized block coordinate descent method." *Advances in neural information processing systems*. 2014.
- 随机加速梯度下降法 : Meng, Qi, et al. "Asynchronous Accelerated Stochastic Gradient Descent." *IJCAI*. 2016.

# 2. 非凸随机优化算法：Ada系列算法

Adam 算法是另一种逐维进行自适应调整步长的算法。Adam 同时引入了以下两个辅助变量分别按照指数衰减形式来累加梯度与梯度的平方：

$$m_{t+1} = \gamma_1 m_t + (1 - \gamma_1) \nabla f(w_t)$$

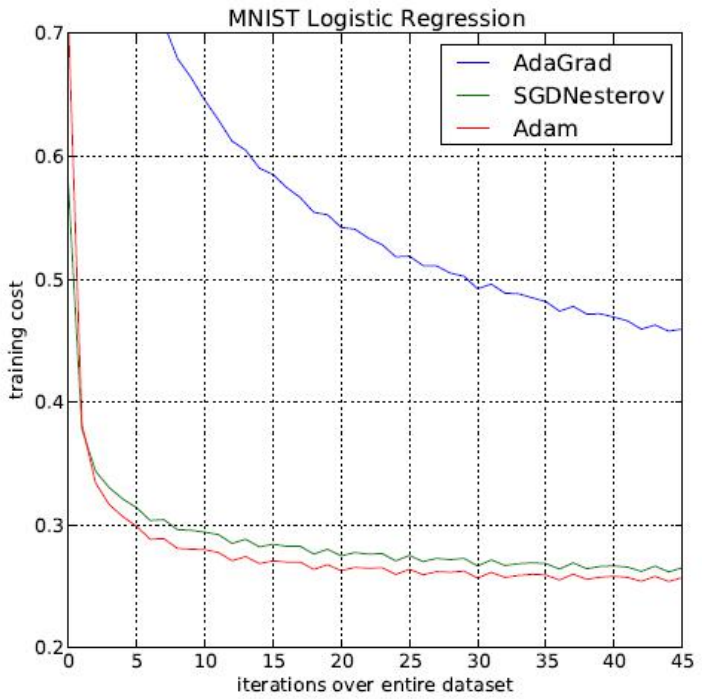
$$g_{t+1} = \gamma_2 g_t + (1 - \gamma_2) (\nabla f(w_t))^2$$

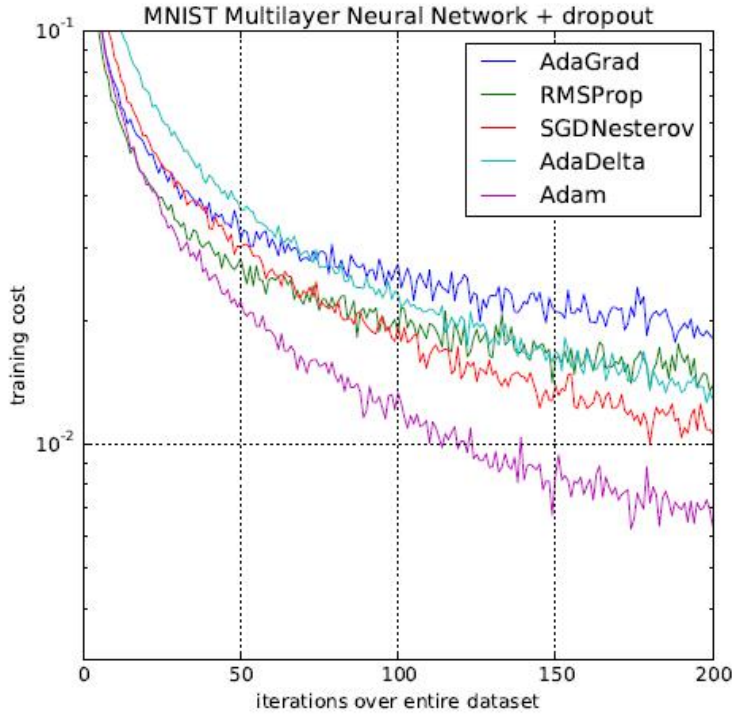
之后对这两个辅助变量的量级进行重整，依照梯度平方累加值调整步长，依照梯度累加值更新模型：

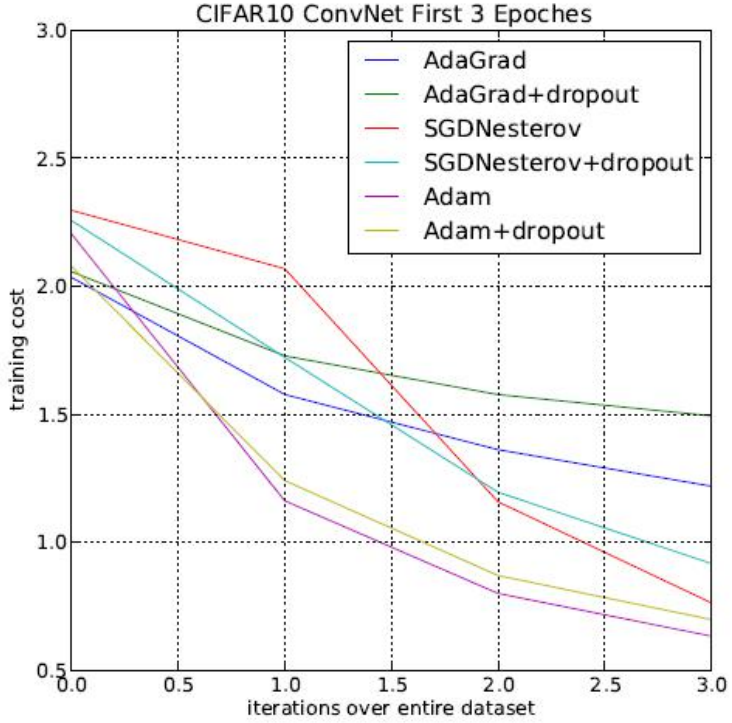
$$\hat{m}_{t+1} = \frac{m_{t+1}}{1 - \gamma_1^{t+1}}, \quad \hat{g}_{t+1} = \frac{g_{t+1}}{1 - \gamma_2^{t+1}}$$

$$w_{t+1} = w_t - \frac{\eta \hat{m}_{t+1}}{\sqrt{\hat{g}_{t+1}} + \varepsilon_0}$$

Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.



# 2. 非凸随机优化算法：非凸理论分析

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left\|\nabla f(w_t)\right\|^2 \quad \text{或者} \quad \min_{t=1,\cdots,T}\mathbb{E}\left\|\nabla f(w_t)\right\|^2$$

1）Ghadimi 和 Lan[44] 证明了随机梯度下降法在非光滑条件下的计算复杂度为 $O\left(n\left(\frac{L}{\varepsilon}+\frac{L\sigma^2}{\varepsilon^2}\right)\right)$，与凸情形下的主阶相同。

2）Reddi 和 Sra 等人[45] 证明了近端随机梯度下降法以及 Frank-Wolfe 算法在非凸条件下的收敛速率和凸情形下相同。

3）Li 和 Lin[46] 分析了加速近端梯度法（APG）在非凸条件下的理论性质，证明了 APG 算法的收敛速率为 $O\left(\frac{1}{\sqrt{\varepsilon}}\right)$，与凸情形下一致，并且证明了 Nesterov 加速法在非凸条件下也能带来多项式阶的加速。

4）Reddi、Zhu、Hazan 等人[18,47] 等人证明了 SVRG 的计算复杂度为 $O\left(n+n^{\frac{2}{3}}\left(\frac{L}{\varepsilon}\right)\right)$，与凸情形下阶数相同。Zhu 和 Yuan[48] 还针对非凸问题设计了 SVRG ++ 算法，该算法使得原始 SVRG 的内循环轮数递增，在非凸问题上取得了比 SVRG 更快的收敛速率。

# 2. 非凸随机优化算法：逃离鞍点问题

1）点$w^*$称为临界点，如果$\nabla f(w^*)=0$.

2）一个临界点$w^*$称为局部极小值点，如果存在$w^*$的一个邻域$U\subseteq\mathcal{W}$，使得$f(w^*)\leqslant f(w)$，$\forall w\in U$。

3）一个临界点$w^*$称为鞍点，如果对于$w^*$的所有邻域，都存在$w$，$v\in\mathcal{W}$，使得$f(w)\leqslant f(w^*)\leqslant f(v)$。临界点$w^*$被称作严格鞍点，如果矩阵$\nabla^2 f(w^*)$的最小特征值严格小于0。

Lee等人[51]证明了随机梯度下降法在一定条件下可以依概率1收敛到局部极小值点，见如下定理。

定理5.8　如果目标函数$f$：$R^d\to R$是二阶连续可微的、具有下界的，并且满足严格鞍点性质，那么带有随机初始化和充分小的步长的梯度下降法可以依概率1收敛到局部极小值点。

# 2. 非凸随机优化算法：等级优化算法

1）通过局部磨光算子将目标函数转变为一个光滑函数，对应粗粒度版本的目标函数。

2）用随机优化算法最小化这个光滑函数。

3）将算法的解作为下一轮优化的起始点，减小磨光力度，返回步骤1。

以下是一种常用的 $\delta$-局部磨光算子。

定义5.3　对于 $L$-Lipschitz 函数 $f$：$R^d \to R$，定义它的 $\delta$-局部磨光算子如下：

$$\hat{f}_\delta(w) = \mathbb{E}_{u \sim B(0,1)}[f(w + \delta u)]$$

其中 $B(0, 1)$ 是以0为中心、1为半径的球。

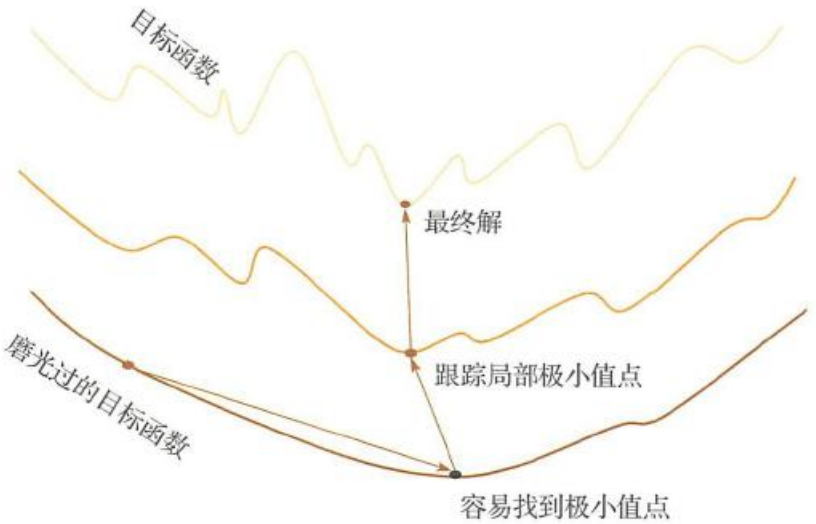



图5.2　等级优化算法示意图