随机优化算法

提纲

监督学习模型

- 假定(a,b)服从概率分布P，其中a为输入，b为标签.
- 例如在自动邮件分类任务中，a表示邮件内容，b表示邮件为正常邮件或垃圾邮件;
- 又例如人脸识别任务中，a表示人脸的图像信息，b表示该人脸属于何人.
- 实际问题中我们不知道真实的概率分布P，而是随机采样得到一个数据集$\mathcal{D}=\{(a_1,b_1),(a_2,b_2),\cdots,(a_N,b_N)\}$. 数据集$\mathcal{D}$对应经验分布

$$\hat{P}=\frac{1}{N}\sum_{n=1}^{N}\delta_{a_i,b_i}$$

其中$\delta_{a,b}$表示(a,b)处的单点分布.

监督学习模型

- 任务是要给定输入a预测标签b，即决定一个最优的函数ϕ使得期望风险$\mathbb{E}[L(\phi(a), b)]$最小，其中$L(\cdot, \cdot)$表示损失函数，函数$\phi$为某个函数空间中的预测函数.
- L的例子如
  - ℓ_2损失函数

$$L(x, y) = \frac{1}{2}\|x - y\|_2^2$$

  - 若$x, y \in \mathbb{R}^d$为概率分布（即各分量和为1的向量），则可定义互熵损失函数

$$L(x, y) = \sum_{i=1}^{d} x_i \log \frac{x_i}{y_i}$$

监督学习模型

实际中为了缩小目标函数的范围，需要将$\phi(\cdot)$参数化为$\phi(\cdot;x)$. ϕ参数化的例子如

- 线性函数

$$\phi(a) = pa + q;$$

  线性函数的参数为p, q.

- 深度神经网络

$$\begin{aligned}
&\phi_0(a) = a \\
&\hat{\phi}_l(a) = W_l\phi_{l-1}(a) + b_h, \quad \phi_l(a) = \sigma(\phi_l(a)) \\
&\phi(a) = \hat{\phi}_L(a).
\end{aligned}$$

  其中$\sigma(\cdot)$为非线性激活函数. 深度神经网络的参数为W_l与b_l.

监督学习模型

- 用经验风险来近似期望风险，即要求解下面的极小化问题：

$$\min_x \quad \frac{1}{N}\sum_{i=1}^{N} L(\phi(a_i;x),b_i) = \mathbb{E}_{(a,b)\sim\hat{P}}[L(\phi(a;x),b)]. \tag{1}$$

- 记

$$f_i(x) = L(\phi(a_i;x),b_i)$$

则只需考虑如下随机优化问题：

$$\min_{x\in\mathbb{R}^n} \quad f(x) \overset{\text{def}}{=\!=} \frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{2}$$

问题(2)也称为随机优化问题的有限和形式.

- 由于数据规模巨大，计算目标函数的梯度非常困难，但是通过采样的方式只计算部分样本的梯度来进行梯度下降，也能达到非常好的数值表现，而每步的运算量却得到了极大的减小.

提纲

梯度下降算法

- 下面为了讨论方便，先假设(2)中每一个$f_i(x)$是凸的、可微的.
- 可以运用梯度下降算法

$$x^{k+1} = x^k - \alpha_k \nabla f(x^k), \tag{3}$$

来求解原始的优化问题.

- 在迭代格式(3)中，

$$\nabla f(x^k) = \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(x^k).$$

要计算这个梯度必须计算出所有的$\nabla f_i(x^k)$然后将它们相加.

- 然而在机器学习中，采集到的样本量是巨大的，因此计算$\nabla f(x^k)$需要非常大的计算量. 使用传统的梯度法求解机器学习问题并不是一个很好的做法.

随机梯度下降算法(SGD)

- SGD的基本迭代格式为

$$x^{k+1} = x^k - \alpha_k \nabla f_{s_k}(x^k), \tag{4}$$

其中s_k是从$\{1, 2, \cdots, N\}$中随机等可能地抽取的一个样本，α_k称为步长. 在机器学习和深度学习领域中，更多的时候被称为学习率(learning rate).

- 通过对比(3)式和(4)式可知，随机梯度算法不去计算全梯度$\nabla f(x^k)$，而是从众多样本中随机抽出一个样本s_i，然后仅仅计算这个样本处的梯度$\nabla f_{s_k}(x^k)$，以此作为$\nabla f(x^k)$的近似.
- 在全梯度$\nabla f(x^k)$的表达式中含系数$1/N$，而迭代格式(4)中不含$1/N$. 这是因为我们要保证随机梯度的条件期望恰好是全梯度，即

$$\mathbb{E}_{s_k}[\nabla f_{s_k}(x^k)|x^k] = \nabla f(x^k).$$

小批量随机梯度法

- 实际计算中每次只抽取一个样本s_k的做法比较极端，常用的形式是小批量（mini-batch）随机梯度法.
- 每次迭代中，随机选择一个元素个数很少的集合$\mathcal{I}_k \subset \{1, 2, \cdots, N\}$，然后执行迭代格式

$$x^{k+1} = x^k - \frac{\alpha_k}{|\mathcal{I}_k|} \sum_{s \in \mathcal{I}_k} \nabla f_s(x^k),$$

其中$|\mathcal{I}_k|$表示$\mathcal{I}_k$中的元素个数.

随机次梯度法

- 当$f_i(x)$是凸函数但不一定可微时，我们可以用$f_i(x)$的次梯度代替梯度进行迭代. 这就是随机次梯度算法.
- 它的迭代格式为

$$x^{k+1} = x^k - \alpha_k g^k, \tag{5}$$

其中α_k为步长，$g^k \in \partial f_{s_k}(x^k)$为随机次梯度，其期望为真实的次梯度.

动量方法

- 传统的梯度法在问题比较病态时收敛速度非常慢，随机梯度下降法也有类似的问题．为了克服这一缺陷，人们提出了动量方法（momentum），其思想是在算法迭代时一定程度上保留之前更新的方向，同时利用当前计算的梯度调整最终的更新方向．
- 动量方法的具体迭代格式如下：

$$v^{k+1} = \mu_k v^k - \alpha_k \nabla f_{s_k}(x^k), \tag{6}$$

$$x^{k+1} = x^k + v^{k+1}. \tag{7}$$

在计算当前点的随机梯度$\nabla f_{s_i}(x^k)$后，我们并不是直接将其更新到变量x^k上，而是将其和上一步更新方向v^k做线性组合来得到新的更新方向v^{k+1}.

动量方法

- 由动量方法迭代格式立即得出当$\mu_k = 0$时该方法退化成随机梯度下降法. 在动量方法中，参数μ_k的范围是$[0, 1)$，通常取$\mu_k \geq 0.5$，其含义为迭代点带有较大惯性，每次迭代会在原始迭代方向的基础上做一个小的修正.
- 在普通的梯度法中，每一步迭代只用到了当前点的梯度估计，动量方法的更新方向还使用了之前的梯度信息.
- 当许多连续的梯度指向相同的方向时，步长就会很大，这从直观上看也是非常合理的.

动量方法

图1比较了梯度法和动量方法的表现. 可以看到普通梯度法生成的点列会在椭圆的短轴方向上来回移动，而动量方法生成的点列更快收敛到了最小值点.

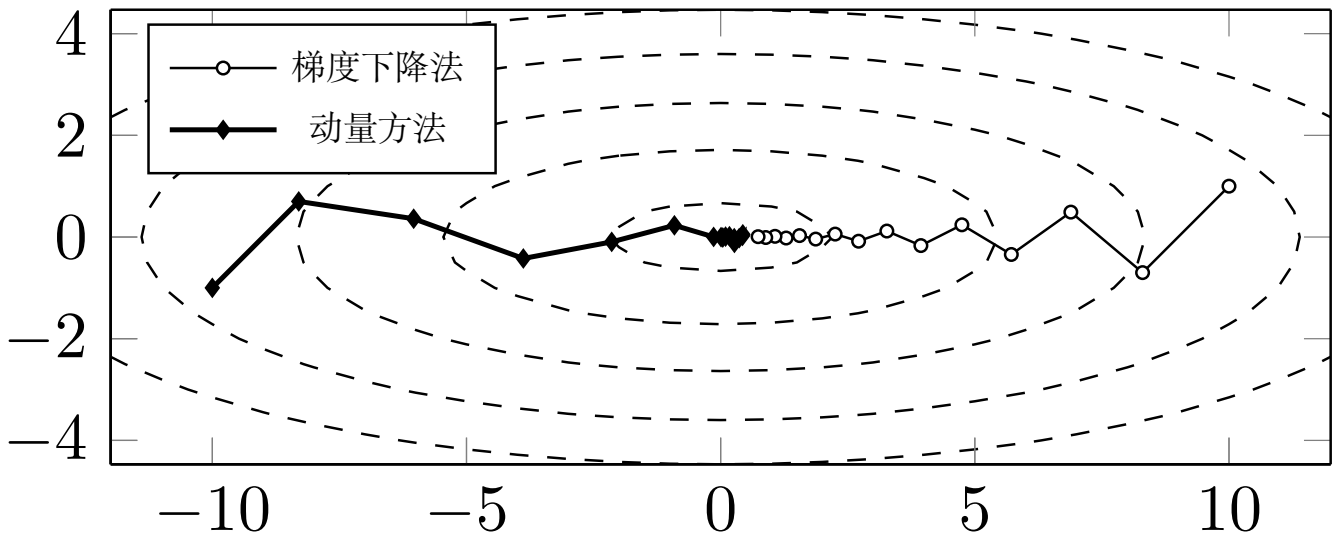

Figure: 动量方法在海瑟矩阵病态条件下的表现

Nesterov加速算法

- 假设$f(x)$为光滑的凸函数. 针对凸问题的Nesterov加速算法为

$$y^k = x^k + \mu_k(x^k - x^{k-1})$$
$$x^{k+1} = y^k - \alpha_k \nabla f(y^k)$$

- 针对光滑问题的Nesterov加速算法迭代的随机版本为

$$y^{k+1} = x^k + \mu_k(x^k - x^{k-1}), \tag{8}$$
$$x^{k+1} = y^{k+1} - \alpha_k \nabla f_{s_k}(y^{k+1}), \tag{9}$$

其中$\mu_k = \frac{k-1}{k+2}$，步长α_k是一个固定值或者由线搜索确定.

- 可以看出，二者的唯一区别为随机版本将全梯度$\nabla f(y^k)$替换为随机梯度$\nabla f_{s_k}(y^{k+1})$.

Nesterov加速算法与动量方法的联系

- 若在第k步迭代引入速度变量$v^k = x^k - x^{k-1}$，再合并原始Nesterov加速算法的两步迭代可以得到

$$x^{k+1} = x^k + \mu_k(x^k - x^{k-1}) - \alpha_k \nabla f_k(x^k + \mu_k(x^k - x^{k-1})).$$

- 定义有关v^{k+1}的迭代式

$$v^{k+1} = \mu_k v^k - \alpha_k \nabla f_k(x^k + \mu_k v^k),$$

- 于是得到关于x^k和v^k的等价迭代：

$$\begin{aligned} v^{k+1} &= \mu_k v^k - \alpha_k \nabla f_{s_k}(x^k + \mu_k v^k), \\ x^{k+1} &= x^k + v^{k+1}. \end{aligned}$$

- 二者的主要差别在梯度的计算上．Nesterov加速算法先对点施加速度的作用，再求梯度，可以理解为对标准动量方法做了校正．

AdaGrad

- 在一般的随机梯度法中，调参是一个很大的难点．我们希望算法能在运行的过程中，根据当前情况自发地调整参数．
- 对无约束光滑凸优化问题，点x是问题的解等价于该点处梯度为零向量．但梯度的每个分量收敛到零的速度是不同的．传统梯度算法只有一个统一的步长α_k来调节每一步迭代，它没有针对每一个分量考虑．
- 当梯度的某个分量较大时，可以推断出在该方向上函数变化比较剧烈，要用小步长；当梯度的某个分量较小时，在该方向上函数比较平缓，要用大步长．AdaGrad就是根据这个思想设计的．

AdaGrad

- 令$g^k = \nabla f_{s_k}(x^k)$，为了记录整个迭代过程中梯度各个分量的累积情况，引入向量

$$G^k = \sum_{i=1}^{k} g^i \odot g^i.$$

从G^k的定义可知G^k的每个分量表示在迭代过程中，梯度在该分量处的累积平方和．当G^k的某分量较大时，我们认为该分量变化比较剧烈，因此应采用小步长，反之亦然．

- 因此AdaGrad的迭代格式为

$$x^{k+1} = x^k - \frac{\alpha}{\sqrt{G^k + \varepsilon \mathbf{1}_n}} \odot g^k, \tag{10}$$

$$G^{k+1} = G^k + g^{k+1} \odot g^{k+1}, \tag{11}$$

这里$\frac{\alpha}{\sqrt{G^k+\varepsilon \mathbf{1}_n}}$中的除法和求根运算都是对向量每个分量分别操作的（下同），α为初始步长，引入$\varepsilon \mathbf{1}_n$这一项是为了防止除零运算．

AdaGrad

- 可以看到AdaGrad的步长大致反比于历史梯度累计值的算术平方根，所以梯度较大时步长下降很快，反之则下降较慢，这样做的效果是在参数空间更平缓的方向上，前后两次迭代的距离较大.
- 在凸优化问题中AdaGrad有比较好的理论性质，但实际应用中也发现在训练深度神经网络模型时，从训练开始就积累梯度平方会导致步长过早或过多减小.

AdaGrad的收敛阶

- 如果在AdaGrad中使用真实梯度$\nabla f(x^k)$，那么AdaGrad也可以看成是一种介于一阶和二阶的优化算法.
- 考虑$f(x)$在点x^k处的二阶泰勒展开：

$$f(x) \approx f(x^k) + \nabla f(x^k)^\top (x - x^k) + \frac{1}{2}(x - x^k)^\top B^k (x - x^k),$$

我们知道选取不同的B^k可以导出不同的优化算法. AdaGrad是使用一个对角矩阵来作为B^k. 具体地，取

$$B^k = \frac{1}{\alpha}\mathrm{Diag}(\sqrt{G^k + \varepsilon \mathbf{1}_n})$$

时导出的算法就是AdaGrad.

RMSProp

- RMSProp（root mean square propagation）是对AdaGrad的一个改进，该方法在非凸问题上可能表现更好. AdaGrad会累加之前所有的梯度分量平方，这就导致步长是单调递减的，因此在训练后期步长会非常小，计算的开销也较大.
- RMSProp提出只需使用离当前迭代点比较近的项，同时引入衰减参数ρ. 具体地，令

$$M^{k+1} = \rho M^k + (1-\rho) g^{k+1} \odot g^{k+1},$$

再对其每个分量分别求根，就得到均方根(root mean square)

$$R^k = \sqrt{M^k + \varepsilon \mathbf{1}_n}, \tag{12}$$

最后将均方根的倒数作为每个分量步长的修正.

RMSProp

- RMSProp迭代格式为：

$$x^{k+1} = x^k - \frac{\alpha}{R^k} \odot g^k, \tag{13}$$

$$M^{k+1} = \rho M^k + (1-\rho) g^{k+1} \odot g^{k+1}. \tag{14}$$

  引入参数ε同样是为了防止分母为0的情况发生. 一般取$\rho = 0.9$, $\alpha = 0.001$.

- 可以看到RMSProp 和AdaGrad 的唯一区别是将G^k替换成了M^k.

AdaDelta

- AdaDelta在RMSProp的基础上，对历史的Δx^k也同样累积平方并求均方根：

$$D^k = \rho D^{k-1} + (1-\rho)\Delta x^k \odot \Delta x^k, \tag{15}$$

$$T^k = \sqrt{D^k + \varepsilon \mathbf{1}_n}, \tag{16}$$

然后使用T^{k-1}和R^k的商对梯度进行校正：

$$\Delta x^k = -\frac{T^{k-1}}{R^k} \odot g^k, \tag{17}$$

$$x^{k+1} = x^k + \Delta x^k. \tag{18}$$

其中T^k和R^k的定义分别为(16) 式和(12)式.

- AdaDelta的特点是步长选择较为保守，同时也改善了AdaGrad步长单调下降的缺陷.

Adam

- Adam选择了一个动量项进行更新：

$$S^k = \rho_1 S^{k-1} + (1-\rho_1) g^k.$$

- 类似RMSProp，Adam也会记录梯度的二阶矩：

$$M^k = \rho_2 M^{k-1} + (1-\rho_2) g^k \odot g^k.$$

- 与原始动量方法和RMSProp的区别是，由于S^k和M^k本身带有偏差，Adam在更新前先对其进行修正：

$$\hat{S}^k = \frac{S^k}{1-\rho_1^k}, \quad \hat{M}^k = \frac{M^k}{1-\rho_2^k},$$

这里ρ_1^k, ρ_2^k分别表示ρ_1, ρ_2的k次方.

- Adam最终使用修正后的一阶矩和二阶矩进行迭代点的更新.

$$x^{k+1} = x^k - \frac{\alpha}{\sqrt{\hat{M}^k + \varepsilon \mathbf{1}_n}} \odot \hat{S}^k.$$

提纲

收敛性假设：凸函数

- 每个$f_i(x)$是闭凸函数，存在次梯度；
- 随机次梯度二阶矩是一致有界的，即存在M，对任意的$x \in \mathbb{R}^n$以及随机下标s_k，有

$$\mathbb{E}_{s_k}[\|g^k\|^2] \leq M^2 < +\infty, \quad g^k \in \partial f_{s_k}(x^k);$$

- 迭代的随机点列$\{x^k\}$处处有界，即$\|x^k - x^*\| \leq R, \forall K$，其中$x^*$是问题(2)的最优解.

引理

我们有如下重要的引理：

引理

在上述假设下，令$\{\alpha_k\}$是任一正步长序列，$\{x^k\}$是由随机次梯度法产生的序列，那么对所有的$K \geq 1$，有

$$\sum_{k=1}^{K} \alpha_k \mathbb{E}[f(x^k) - f(x^*)] \leq \frac{1}{2}\mathbb{E}[\|x^1 - x^*\|^2] + \frac{1}{2}\sum_{k=1}^{K} \alpha_k^2 M^2. \tag{19}$$

引理的证明

- 令$\bar{g}^k = \mathbb{E}[g^k|x^k]$, $\xi^k = g^k - \bar{g}^k$.
- 由随机次梯度法的性质，

$$\bar{g}^k = \mathbb{E}[g^k|x^k] \in \partial f(x^k),$$

- 由次梯度的性质，

$$\langle \bar{g}^k, x^* - x^k \rangle \leq f(x^*) - f(x^k).$$

可以推导得

$$\begin{aligned}
&\|x^{k+1} - x^*\|^2 \\
=& \|x^k - \alpha_k g^k - x^*\|^2 \\
=& \|x^k - x^*\|^2 + 2\alpha_k \langle g^k, x^* - x^k \rangle + \alpha_k^2 \|g^k\|^2 \\
=& \|x^k - x^*\|^2 + 2\alpha_k \langle \bar{g}^k, x^* - x^k \rangle + \alpha_k^2 \|g^k\|^2 + 2\alpha_k \langle \xi^k, x^* - x^k \rangle \\
\leq& \|x^k - x^*\|^2 + 2\alpha_k (f(x^*) - f(x^k)) + \alpha_k^2 \|g^k\|^2 + 2\alpha_k \langle \xi^k, x^* - x^k \rangle.
\end{aligned} \tag{20}$$

- 注意到$\mathbb{E}[\xi^k|x^k]=0$，所以

$$\mathbb{E}[\langle \xi^k, x^*-x^k\rangle]=\mathbb{E}[\mathbb{E}[\langle \xi^k, x^*-x^k\rangle|x_k]]=0.$$

- 对不等式(20)两端求期望就得到

$$\alpha_k\mathbb{E}[f(x^k)-f(x^*)]\leq \frac{1}{2}\mathbb{E}[\|x^k-x^*\|^2]-\frac{1}{2}\mathbb{E}[\|x^{k+1}-x^*\|^2]+\frac{\alpha_k^2}{2}M^2. \quad (21)$$

- 两边对k求和即得证.

随机次梯度算法的收敛性1

根据该引理，我们很容易得到随机次梯度算法在收缩步长下的收敛性.

定理 (随机次梯度算法的收敛性1)

在收敛性假设的条件下，令$A_K = \sum_{i=1}^{K} \alpha_i$，定义$\bar{x}_K = \frac{1}{A_K} \sum_{k=1}^{K} \alpha_k x^k$，则

$$\mathbb{E}[f(\bar{x}_K) - f(x^*)] \leq \frac{R^2 + \sum_{k=1}^{K} \alpha_k^2 M^2}{2 \sum_{k=1}^{K} \alpha_k}. \tag{22}$$

定理的证明

- 由$f(x)$的凸性以及引理1得到

$$\begin{aligned} & A_k \mathbb{E}[f(\bar{x}_K) - f(x^*)] \\ \leq & \sum_{k=1}^{K} \alpha_k \mathbb{E}[f(x^k) - f(x^*)] \\ \leq & \frac{1}{2}\mathbb{E}[\|x^1 - x^*\|^2] + \frac{1}{2}\sum_{k=1}^{K} \alpha_k^2 M^2 \\ = & \frac{R^2 + \sum_{k=1}^{K} \alpha_k^2 M^2}{2} \end{aligned}$$

- 不等式两边同除以A_K得到

$$\mathbb{E}[f(\bar{x}_K) - f(x^*)] \leq \frac{R^2 + \sum\limits_{k=1}^{K} \alpha_k^2 M^2}{2A_K}.$$

随机次梯度算法的收敛性1

- 从定理1可以看到，当

$$\sum_{k=1}^{\infty} \alpha_k = +\infty, \quad \frac{\sum_{k=1}^{K} \alpha_k^2}{\sum_{k=1}^{K} \alpha_k} \to 0$$

时，随机次梯度算法收敛.

- 对一个固定的步长α，不等式(22)右侧有一个不随K递减的常数，因此固定步长随机次梯度算法在函数值取期望意义下是不收敛的，它仅仅能找到一个次优解：

$$\mathbb{E}[f(\bar{x}_K) - f(x^*)] \leq \frac{R^2}{2K\alpha} + \frac{\alpha M^2}{2}.$$

特别地，对于给定的迭代次数K，选取固定步长$\alpha = \frac{R}{M\sqrt{K}}$，可以达到$\mathcal{O}(1/\sqrt{K})$的精度，即

$$\mathbb{E}[f(\bar{x}_K) - f(x^*)] \leq \frac{RM}{\sqrt{K}}.$$

随机次梯度算法的收敛性2

在步长不增的情况下，我们可以得到直接平均意义下的收敛性.

定理 (随机次梯度算法的收敛性2)

在收敛性假设的条件下，令$\{\alpha_k\}$是一个不增的正步长序列，$\bar{x}_K = \frac{1}{K}\sum_{k=1}^{K} x^k$，则

$$\mathbb{E}[f(\bar{x}_K) - f(x^*)] \leq \frac{R^2}{2K\alpha_K} + \frac{1}{2K}\sum_{k=1}^{K}\alpha_k M^2. \tag{23}$$

定理的证明

- 对(21)式两边同除α_k，就有

$$\mathbb{E}[f(x^k)-f(x^*)] \leq \frac{1}{2\alpha_k}\mathbb{E}[\|x^k-x^*\|_2^2] - \frac{1}{2\alpha_k}\mathbb{E}[\|x^{k+1}-x^*\|_2^2] + \frac{\alpha_k}{2}M^2.$$

- 再对k求和，并且利用$f(x)$的凸性和α_k的单调性得

$$\begin{aligned}
\mathbb{E}[f(\bar{x}_K)-f(x^*)] &\leq \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}[f(x^k)-f(x^*)] \\
&\leq \frac{1}{2K}\left(\frac{1}{\alpha_1}\mathbb{E}[\|x^1-x^*\|^2] + \sum_{k=1}^{K}\alpha_k M^2 + \right. \\
&\left. \sum_{k=2}^{K}\left(\frac{1}{\alpha_k}-\frac{1}{\alpha_{k-1}}\right)\mathbb{E}[\|x^k-x^*\|^2]\right) \\
&\leq \frac{R}{2K\alpha_k} + \frac{1}{2K}\sum_{k=1}^{K}\alpha_k M^2.
\end{aligned}$$

随机次梯度算法的收敛性2

- 注意该定理和定理1 的不同之处在于$\bar{x}_K$的定义.
- 通过选取$\mathcal{O}(1/\sqrt{k})$阶数的步长，我们可以得到目标函数的收敛速度为$\mathcal{O}(1/\sqrt{k})$:

推论

在收敛性假设的条件下，令$\alpha_k = \frac{R}{M\sqrt{k}}$，则

$$\mathbb{E}[f(\bar{x}_K) - f(x^*)] \leq \frac{3RM}{2\sqrt{K}}. \tag{24}$$

其中$\bar{x}_K$的定义和定理2 相同.

推论的证明

- 注意到

$$\sum_{k=1}^{K}\frac{1}{\sqrt{k}}\leq\int_{0}^{K}\frac{1}{\sqrt{t}}\mathrm{d}t=2\sqrt{K}.$$

- 将$\alpha_k=\frac{R}{M\sqrt{k}}$代入式(23)就得到

$$\mathbb{E}[f(\bar{x}_K)-f(x^*)]\leq\frac{R^2}{2K\frac{R}{M\sqrt{K}}}+\frac{RM}{2K}2\sqrt{K}=\frac{3RM}{2\sqrt{K}}.$$

随机次梯度算法的收敛性2

- 我们可以发现随机次梯度算法和非随机次梯度算法具有相同的收敛速度——$\mathcal{O}(1/\sqrt{k})$.
- 随机次梯度算法每步的计算代价远小于非随机次梯度，这一定程度上解释了为什么随机算法在一些问题中的表现要远远好于非随机算法.

随机次梯度算法的收敛性3

下面主要讨论随机次梯度算法在依概率意义下的收敛性和收敛速度.

定理

选择上述推论中的步长α_k，使得$\mathbb{E}[f(\bar{x}_K)-f(x^*)]\to 0$，那么我们有依概率收敛$f(\bar{x}_K)-f(x^*)\xrightarrow{P}0\ (K\to\infty)$，即对任意的$\varepsilon>0$，都有

$$\lim_{K\to\infty} P(f(\bar{x}_K)-f(x^*)\geq\varepsilon)=0. \tag{25}$$

- 由马尔可夫不等式立即得到

$$P(f(\bar{x}_K)-f(x^*)\geq\varepsilon)\leq\frac{1}{\varepsilon}\mathbb{E}[f(\bar{x}_K)-f(x^*)]\to 0.$$

随机变量的收敛性

- 随机变量本质上是定义在样本空间上的函数. 和函数列一样，我们可以定义一列随机变量的极限.
- 称随机变量序列$\{X_n\}_{n=1}^{\infty}$几乎必然收敛到X，如果

$$P\left(\lim_{n\to\infty} X_n = X\right) = 1;$$

- 称随机变量序列$\{X_n\}_{n=1}^{\infty}$依概率收敛到X，如果

$$\lim_{n\to\infty} P(|X_n - X| \geq \varepsilon) = 0.$$

随机次梯度算法的收敛性3

定理 (随机次梯度算法的收敛性3)

在假设收敛性假设的条件下，进一步假设对于所有的随机次梯度g，有$\|g\| \leq M$. 那么对任意的$\varepsilon > 0$，

$$f(\bar{x}_K) - f(x^*) \leq \frac{R^2}{2K\alpha_K} + \frac{1}{2K}\sum_{k=1}^{K}\alpha_k M^2 + \frac{RM}{\sqrt{K}}\varepsilon \tag{26}$$

以大于等于$1 - e^{-\frac{1}{2}\varepsilon^2}$的概率成立，其中步长列$\{\alpha_k\}$是单调不增序列，$\bar{x}_K$的定义和定理2 中的定义相同.

鞅差序列

定义

设$\{X_n\}_{n=1}^{\infty}$与$\{Z_n\}_{n=1}^{\infty}$为$(\Omega, \mathcal{F}, P)$上的随机过程. 如果$\forall n \in \mathbb{N}_+$,

1. $\mathbb{E}[|X_n|] < +\infty$;
2. X_n属于$Z_1, \ldots, Z_n$生产的σ-代数;
3. $\mathbb{E}[X_{n+1}|Z_1, \ldots, Z_n] = 0$.

则称$\{X_n\}_{n=1}^{\infty}$为关于$\{Z_n\}_{n=1}^{\infty}$的鞅差序列. 特别地, 若$\{Z_n\}_{n=1}^{\infty} = \{X_n\}_{n=1}^{\infty}$, 则称$\{X_n\}_{n=1}^{\infty}$为鞅差序列.

- 鞅差序列的例子如: 设$\{Z_n\}_{n=1}^{\infty}$为一维简单随机游走, $\{X_n\}_{n=1}^{\infty}$定义为$X_n = Z_n - Z_{n-1}$(补充定义$Z_0 = 0$).
- 由一维简单随机游走的性质, X_n以$1/2$的概率为1, 以$1/2$的概率为-1. 因此$\mathbb{E}[|X_n|] = 1 < +\infty$;
- X_n由Z_{n-1}及Z_n完全决定, 因此属于$Z_1, \ldots, Z_n$生成的σ-代数;
- 由一维简单随机游走的性质, $X_{n+1} = Z_{n+1} - Z_n$独立于$Z_1, \ldots, Z_n$, 所以$\mathbb{E}[X_{n+1}|Z_1, \ldots, Z_n] = \mathbb{E}[X_{n+1}] = 0$.

Azuma-Hoeffding不等式

下面给出Azuma-Hoeffding不等式，即鞅版本的Hoeffding不等式：

引理

设$\{X_n\}_{n=1}^{\infty}$为鞅差序列，且$\forall n \in \mathbb{N}_+, |X_n| \leq B$. 则$\forall t > 0$，

$$P\left(\sum_{i=1}^{n} X_i \geq t\right) \leq \exp\left(-\frac{2t^2}{nB^2}\right);$$

$$P\left(\sum_{i=1}^{n} X_i \leq -t\right) \leq \exp\left(-\frac{2t^2}{nB^2}\right).$$

定理的证明

- 令$\bar{g}^k = \mathbb{E}[g^k|x^k]$, $\xi^k = g^k - \bar{g}^k$. 由(20)式的推导过程我们已经得到

$$f(x^k) - f(x^*) \leq \frac{1}{2\alpha_k}\|x^k - x^*\|^2 - \frac{1}{2\alpha_k}\|x^{k+1} - x^*\|^2 + \frac{\alpha_k}{2}\|g^k\|^2 + \langle \xi^k, x^* - x^k \rangle.$$

- 利用$f(x)$的凸性与α_k的单调性有

$$\begin{aligned} f(\bar{x}_K) - f(x^*) &\leq \frac{1}{K}\sum_{k=1}^{K} f(x^k) - f(x^*) \\ &\leq \frac{R^2}{2K\alpha_K} + \frac{1}{2K}\sum_{k=1}^{K}\alpha_k\|g^k\|^2 + \frac{1}{K}\sum_{k=1}^{K}\langle \xi^k, x^* - x^k \rangle \\ &\leq \frac{R^2}{2K\alpha_K} + \frac{1}{2K}\sum_{k=1}^{K}\alpha_k M^2 + \frac{1}{K}\sum_{k=1}^{K}\langle \xi^k, x^* - x^k \rangle. \end{aligned}$$

- 令

$$\omega = \frac{R^2}{2K\alpha_K} + \frac{1}{2K}\sum_{k=1}^{K}\alpha_k M^2$$

得到

$$P(f(\bar{x}_K) - f(x^*) - \omega \geq t) \leq P\left(\frac{1}{K}\sum_{k=1}^{K}\langle \xi^k, x^* - x^k\rangle \geq t\right). \qquad (27)$$

- 设$Z^k = (x^1, x^2, \cdots, x^{k+1})$. 因为

$$\mathbb{E}[\xi^k | Z^{k-1}] = \mathbb{E}[\xi^k | x^k] = 0, \quad \mathbb{E}[x^k | Z^{k-1}] = x^k,$$

我们知道$\langle \xi^k, x^* - x^k\rangle$是一个鞅差序列.

- 同时由

$$\|\xi^k\|_2 = \|g^k - \overline{g}^k\|_2 \leq 2M$$

推出

$$|\langle \xi^k, x^* - x^k \rangle| \leq \|\xi^k\| \|x^* - x^k\|_2 \leq 2MR$$

即$\langle \xi_k, x^* - x_k \rangle$有界.

- 由Azuma-Hoeffding不等式得到

$$P\left(\frac{1}{K} \sum_{k=1}^{K} \langle \xi^k, x^* - x^k \rangle \geq t \right) \leq \exp\left(-\frac{Kt^2}{2M^2R^2} \right)$$

- 将$t = \frac{MR\varepsilon}{\sqrt{K}}$代入，有

$$P\left(\frac{1}{K} \sum_{k=1}^{K} \langle \xi^k, x^* - x^k \rangle \geq \frac{MR\varepsilon}{\sqrt{K}} \right) \leq \exp\left(-\frac{\varepsilon^2}{2} \right).$$

结合(27)式，定理得证.

随机次梯度算法的收敛性3

- 如果取$\alpha_k = \frac{R}{\sqrt{k}M}$，并令$\delta = e^{-\frac{1}{2}\varepsilon^2}$，就有

$$P\left(f(\bar{x}_K) - f(x^*) \leq \frac{3RM}{2\sqrt{K}} + \frac{RM\sqrt{2\ln 1/\delta}}{\sqrt{K}}\right) \geq 1 - \delta. \tag{28}$$

- 可以看到除一个很小的概率外，函数值以$\mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$的速度收敛.

收敛性假设

对问题(2)使用迭代算法(4)时，

1. $f(x)$是可微函数，每个$f_i(x)$梯度存在；
2. $f(x)$是梯度利普希茨连续的，相应常数为L；
3. $f(x)$是强凸函数，强凸参数为μ；
4. 随机梯度二阶矩是一致有界的，即存在M，对任意的$x \in \mathbb{R}^n$以及随机下标s^k，有

$$\mathbb{E}_{s_k}[\|\nabla f_{s_k}(x)\|^2] \leq M^2 < +\infty.$$

随机梯度算法的收敛性：强凸函数

下面的定理将给出随机梯度算法在固定步长下的收敛性分析.

定理 (随机梯度算法的收敛性)

在收敛性假设的条件下，定义$\Delta_k = \|x^k - x^*\|$. 对固定的步长$\alpha_k = \alpha$, $0 < \alpha < \frac{1}{2\mu}$，有

$$\mathbb{E}[f(x^{K+1}) - f(x^*)] \leq \frac{L}{2}\mathbb{E}[\Delta_{K+1}^2] \leq \frac{L}{2}\left[(1-2\alpha\mu)^K \Delta_1^2 + \frac{\alpha M^2}{2\mu}\right]. \quad (29)$$

定理的证明

- 根据随机梯度算法的更新公式，

$$\begin{aligned}\Delta_{k+1}^2 &= \|x^{k+1} - x^*\|^2 = \|x^k - \alpha_k \nabla f_{s_k}(x^k) - x^*\|^2 \\ &= \|x^k - x^*\|^2 - 2\alpha_k \langle \nabla f_{s_k}(x^k), x^k - x^* \rangle + \alpha_k^2 \|\nabla f_{s_k}(x^k)\|^2 \\ &= \Delta_k^2 - 2\alpha_k \langle \nabla f_{s_k}(x^k), x^k - x^* \rangle + \alpha_k^2 \|\nabla f_{s_k}(x^k)\|^2,\end{aligned}$$

- 由条件期望的性质$\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X|Y]]$，有

$$\begin{aligned}&\mathbb{E}_{s_1,s_2,\cdots,s_k}[\langle \nabla f_{s_k}(x^k), x^k - x^* \rangle] \\ =\ &\mathbb{E}_{s_1,s_2,\cdots,s_{k-1}}[\mathbb{E}_{s_k}[\langle \nabla f_{s_k}(x^k), x^k - x^* \rangle | s_1, \cdots, s_{k-1}]] \\ =\ &\mathbb{E}_{s_1,s_2,\cdots,s_{k-1}}[\langle \mathbb{E}_{s_k}[\nabla f_{s_k}(x_k) | s_1, s_2, \cdots, s_{k-1}], x^k - x^* \rangle] \\ =\ &\mathbb{E}_{s_1,s_2,\cdots,s_{k-1}}[\langle \nabla f(x^k), x^k - x^* \rangle] \\ =\ &\mathbb{E}_{s_1,s_2,\cdots,s_k}[\langle \nabla f(x^k), x^k - x^* \rangle].\end{aligned}$$

- 根据强凸函数的单调性，

$$\left\langle \nabla f(x^k), x^k - x^* \right\rangle = \left\langle \nabla f(x^k) - \nabla f(x^*), x^k - x^* \right\rangle \geq \mu \|x^k - x^*\|^2.$$

- 由随机梯度二阶矩的一致有界性，

$$\mathbb{E}_{s_1,s_2,\cdots,s_k}[\Delta_{k+1}^2] \leq (1 - 2\alpha\mu)\mathbb{E}_{s_1,s_2,\cdots,s_k}[\Delta_k^2] + \alpha^2 M^2. \tag{30}$$

- 对k做归纳，就得到

$$\mathbb{E}_{s_1,s_2,\cdots,s_K}[\Delta_{K+1}^2] \leq (1 - 2\alpha\mu)^K \Delta_1^2 + [1 - (1 - 2\alpha\mu)^K] \frac{\alpha M^2}{2\mu}.$$

由$0 < 2\alpha\mu < 1$可知

$$\mathbb{E}_{s_1,s_2,\cdots,s_K}[\Delta_{K+1}^2] \leq (1 - 2\alpha\mu)^K \Delta_1^2 + \frac{\alpha M^2}{2\mu}. \tag{31}$$

- 利用梯度L-利普希茨连续函数的二次上界，可以得到

$$f(x^{K+1}) - f(x^*) \leq \left\langle \nabla f(x^*), x^{K+1} - x^* \right\rangle + \frac{L}{2}\|x^{K+1} - x^*\|^2.$$

- 利用$\nabla f(x^*) = 0$并对上式左右两边取期望可得

$$\mathbb{E}[f(x^{K+1}) - f(x^*)] \leq \frac{L}{2}\mathbb{E}[\Delta_{K+1}^2] \leq \frac{L}{2}\left[(1 - 2\alpha\mu)^K \Delta_1^2 + \frac{\alpha M^2}{2\mu}\right].$$

随机梯度算法的收敛性

下面的定理表明，如果设置递减的步长，收敛阶可以达到$\mathcal{O}(1/K)$.

定理

随机梯度算法的收敛速度 在上述定理的结果中，在收敛性假设的条件下，取递减的步长

$$\alpha_k = \frac{\beta}{k+\gamma},$$

其中$\beta > \frac{1}{2\mu}, \gamma > 0$，使得$\alpha_1 \leq \frac{1}{2\mu}$，那么对于任意的$k \geq 1$，都有

$$\mathbb{E}[f(x^k) - f(x^*)] \leq \frac{L}{2}\mathbb{E}[\Delta_k^2] \leq \frac{L}{2}\frac{v}{\gamma+k}, \tag{32}$$

这里

$$v = \max\left\{\frac{\beta^2 M^2}{2\beta\mu - 1}, (\gamma+1)\Delta_1^2\right\}.$$

定理的证明

- 之前的定理已经证明了

$$\mathbb{E}_{s_1,s_2,\cdots,s_k}[\Delta_{k+1}^2] \leq (1-2\alpha_k\mu)\mathbb{E}_{s_1,s_2,\cdots,s_k}[\Delta_k^2] + \alpha_k^2 M^2.$$

- 用数学归纳法证明(32)式. 由v的定义知$k=1$时(32)式成立.
- 现假设该式对k成立，定义$\hat{k}=\gamma+k$，则$\alpha_k=\beta/\hat{k}$. 由归纳假设，

$$\begin{aligned}\mathbb{E}[\Delta_{k+1}^2] &\leq \left(1-\frac{2\beta\mu}{\hat{k}}\right)\frac{v}{\hat{k}}+\frac{\beta^2M^2}{\hat{k}^2}\\ &= \frac{\hat{k}-1}{\hat{k}^2}v-\frac{2\beta\mu-1}{\hat{k}^2}v+\frac{\beta^2M^2}{\hat{k}^2}\\ &\leq \frac{v}{\hat{k}+1}\end{aligned}$$

最后一个不等式用到了v的定义. 所以(32)式对$k+1$也成立.

随机梯度算法的收敛性

上述定理表明对于强凸函数，随机梯度下降法的收敛速度可以达到$\mathcal{O}(1/K)$. 对于一般的凸函数随机梯度算法也有一定的收敛性，为此我们在下表比较随机算法和普通算法的复杂度.

Table: 梯度下降法的算法复杂度

	f凸(次梯度算法)	f可微强凸	f可微强凸且L-光滑
随机算法	$\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon}\right)$
普通算法	$\mathcal{O}\left(\frac{N}{\varepsilon^2}\right)$	$\mathcal{O}\left(\frac{N}{\varepsilon}\right)$	$\mathcal{O}\left(N\ln\left(\frac{1}{\varepsilon}\right)\right)$

提纲

梯度下降法与随机梯度下降法的比较

下面分析梯度下降法与随机梯度下降法的主要区别：

- 在强凸性假设下，对梯度下降法有

$$\begin{aligned}\Delta_{k+1}^2 &= \|x^{k+1}-x^*\|^2 = \|x^k-\alpha\nabla f(x^k)-x^*\|^2 \\ &= \Delta_k^2 - 2\alpha\langle\nabla f(x^k), x^k-x^*\rangle + \alpha^2\|\nabla f(x^k)\|^2 \\ &\le (1-2\alpha\mu)\Delta_k^2 + \alpha^2\|\nabla f(x^k)\|_2^2 \qquad (\mu\text{-强凸}) \\ &\le (1-2\alpha\mu+\alpha^2L^2)\Delta_k^2. \qquad (L\text{-光滑})\end{aligned} \tag{33}$$

梯度下降法与随机梯度下降法的比较

- 对随机梯度下降法，利用条件期望的性质有

$$
\begin{aligned}
\mathbb{E}[\Delta_{k+1}^2] &= \mathbb{E}[\|x^{k+1}-x^*\|_2^2] = \mathbb{E}[\|x^k-\alpha\nabla f_{s_k}(x^k)-x^*\|]^2 \\
&= \mathbb{E}[\Delta_k^2] - 2\alpha\mathbb{E}[\langle\nabla f_{s_k}(x^k), x^k-x^*\rangle] + \alpha^2\mathbb{E}[\|\nabla f_{s_k}(x^k)\|^2] \\
&= \mathbb{E}[\Delta_k^2] - 2\alpha\mathbb{E}[\langle\nabla f(x^k), x^k-x^*\rangle] + \alpha^2\mathbb{E}[\|\nabla f_{s_k}(x^k)\|^2] \\
&\le (1-2\alpha\mu)\mathbb{E}[\Delta_k^2] + \alpha^2\mathbb{E}[\|\nabla f_{s_k}(x^k)\|^2] \qquad (\mu\text{-强凸}) \\
&= (1-2\alpha\mu)\mathbb{E}[\Delta_k^2] + \alpha^2\mathbb{E}[\|\nabla f_{s_k}(x^k)-\nabla f(x^k)+\nabla f(x^k)\|^2] \\
&\le \underbrace{(1-2\alpha\mu+\alpha^2L^2)\mathbb{E}[\Delta_k^2]}_{A} + \underbrace{\alpha^2\mathbb{E}[\|\nabla f_{s_k}(x^k)-\nabla f(x^k)\|^2]}_{B}.
\end{aligned}
\tag{34}
$$

- 可以看到两种算法的主要差别就在B项上，也就是梯度估计的某种方差．它导致了随机梯度算法只能有$\mathcal{O}(1/k)$的收敛速度．

方差减小技术

- 在许多机器学习的应用中，随机梯度算法的收敛速度更快一些.
- 这主要是因为许多应用对解的精度要求不太高，而在开始部分方差较小，即有$B \ll A$，那么我们会观察到近似Q-线性收敛速度；
- 而随着迭代步数增多，方差增大，最终的收敛速度为$\mathcal{O}(1/k)$.
- 为了能获得比较快的渐进收敛速度，我们的主要目标即减少方差项B. 下面介绍三种减小方差的算法：
  - SAG (stochastic average gradient)
  - SAGA
  - SVRG (stochastic variance reduced gradient)

SAG算法

- 当迭代接近收敛时，上一步的随机梯度也是当前迭代点处梯度的一个很好的估计. 随机平均梯度法(SAG)就是基于这一想法.
- 在迭代中，SAG算法记录所有之前计算过的随机梯度，再与当前新计算的随机梯度求平均，最终作为下一步的梯度估计.
- 具体来说，SAG算法在内存中开辟了存储N个随机梯度的空间

$$[g_1^k, g_2^k, \cdots, g_N^k],$$

分别用于记录和第i个样本相关的最新的随机梯度. 在第k步更新时，若抽取的样本点下标为s_k，则计算随机梯度后将$g_{s_k}^k$的值更新为当前的随机梯度值，而其他未抽取到的下标对应的g_i^k保持不变. 每次SAG 算法更新使用的梯度方向是所有g_i^k的平均值.

SAG算法

- SAG算法的迭代格式为

$$x^{k+1} = x^k - \frac{\alpha_k}{N} \sum_{i=1}^{N} g_i^k$$

其中g_i^k的更新方式为

$$g_i^k = \begin{cases} \nabla f_{s_k}(x^k), & i = s_k, \\ g_i^{k-1}, & \text{其他}, \end{cases} \tag{35}$$

这里s_k是第k次迭代随机抽取的样本.

- 由g_i^k的更新方式不难发现，每次迭代时只有一个g_i^k的内容发生了改变. 因此SAG迭代公式还可以写成

$$x^{k+1} = x^k - \alpha_k \left(\frac{1}{N} (\nabla f_{s_k}(x^k) - g_{s_k}^{k-1}) + \frac{1}{N} \sum_{i=1}^{N} g_i^{k-1} \right), \tag{36}$$

SAG算法

- $\{g_i^k\}$的初值可简单地取为0或中心化的随机梯度向量，
- SAG算法每次使用的随机梯度的条件期望并不是真实梯度$\nabla f(x^k)$，但随着迭代进行，随机梯度的期望和真实梯度的偏差会越来越小.

SAG算法的收敛性

定理 (SAG算法的收敛性)

在强凸性收敛性假设的条件下，取固定步长$\alpha_k = \frac{1}{16L}$，g_i^k的初值取为0，则对任意的k，有

$$\mathbb{E}[f(x^k)] - f(x^*) \leq \left(1 - \min\left\{\frac{\mu}{16L}, \frac{1}{8N}\right\}\right)^k C_0,$$

其中常数C_0为与k无关的常数.

- 上述定理表明SAG算法确实有Q-线性收敛速度.
- SAG算法的缺点在于需要存储N个梯度向量，当样本量N很大时，这是一个很大的开销. 因此SAG算法在实际中很少使用.

SAGA算法

- SAGA算法的迭代方式为

$$x^{k+1} = x^k - \alpha_k \left(\nabla f_{s_k}(x^k) - g_{s_k}^{k-1} + \frac{1}{N} \sum_{i=1}^{N} g_i^{k-1} \right). \tag{37}$$

- 对比(36)式可以发现，SAGA算法去掉了$\nabla f_{s_k}(x^k) - g_{s_k}^{k-1}$前面的系数$1/N$. 可以证明每次迭代使用的梯度方向都是无偏的，即

$$\mathbb{E}\left[\nabla f_{s_k}(x^k) - g_{s_k}^{k-1} + \frac{1}{N} \sum_{i=1}^{N} g_i^{k-1} \,\middle|\, x^k \right] = \nabla f(x^k).$$

SAGA算法的收敛性

SAGA算法同样有Q-线性收敛速度：

定理 (SAGA算法的收敛性)

在强凸性收敛性假设的条件下，取固定步长$\alpha_k = \frac{1}{2(\mu N+L)}$. 定义$\Delta_k = \|x^k - x^*\|$，则对任意的$k \geq 1$有

$$\mathbb{E}[\Delta_k^2] \leq \left(1 - \frac{\mu}{2(\mu N + L)}\right)^k \left(\Delta_1^2 + \frac{N(f(x^1) - f(x^*))}{\mu N + L}\right). \tag{38}$$

如果强凸的参数μ是未知的，也可以取$\alpha = \frac{1}{3L}$，有类似的收敛结果.

SVRG算法

- 与SAG算法和SAGA算法不同，SVRG算法通过周期性缓存全梯度的方法来减小方差.
- 具体做法是在随机梯度下降方法中，每经过m次迭代就设置一个检查点，计算一次全梯度，在之后的m次迭代中，将这个全梯度作为参考点来达到减小方差的目的.
- 令$\tilde{x}^j$是第j个检查点，则我们需要计算点$\tilde{x}^j$处的全梯度

$$\nabla f(\tilde{x}^j) = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(\tilde{x}^j),$$

在之后的迭代中使用方向v^k作为更新方向：

$$v^k = \nabla f_{s_k}(x^k) - (\nabla f_{s_k}(\tilde{x}^j) - \nabla f(\tilde{x}^j)), \tag{39}$$

其中$s_k \in \{1, 2, \cdots, N\}$是随机选取的一个样本.

SVRG算法

- 注意到给定$s_1, s_2, \cdots, s_{k-1}$时$x^k, \tilde{x}^j$均为定值，由$v^k$的表达式可知

$$\begin{aligned}
&\mathbb{E}[v^k | s_1, s_2, \cdots, s_{k-1}] \\
=& \mathbb{E}[\nabla f_{s_k}(x^k) | x^k] - \mathbb{E}[\nabla f_{s_k}(\tilde{x}^j) - \nabla f(\tilde{x}^j) | s_1, s_2, \cdots, s_{k-1}] \\
=& \nabla f(x^k) - 0 = \nabla f(x^k),
\end{aligned}$$

- 公式(39)有简单的直观理解：我们希望用$\nabla f_{s_k}(\tilde{x}^j)$去估计$\nabla f(\tilde{x}^j)$，那么$\nabla f_{s_k}(\tilde{x}^j) - \nabla f(\tilde{x}^j)$就可以看作梯度估计的误差，所以在每一步随机梯度迭代用该项来对$\nabla f_{s_k}(x^k)$做一个校正.

SVRG算法

- 假设

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \quad i = 1, 2, \cdots, N.$$

- 令$y = \tilde{x}^j$，x^*为$f(x)$的最小值点，$\Delta_k = \|x^k - x^*\|$，则

$$\begin{aligned}
&\mathbb{E}\left[\|v^k\|^2\right] = \mathbb{E}\left[\|\nabla f_{s_k}(x^k) - (\nabla f_{s_k}(y) - \nabla f(y))\|^2\right] \\
&= \mathbb{E}\left[\|\nabla f_{s_k}(x^k) - \nabla f_{s_k}(y) + \nabla f(y) + \nabla f_{s_k}(x^*) - \nabla f_{s_k}(x^*)\|^2\right] \\
&\leq 2\mathbb{E}\left[\|\nabla f_{s_k}(x^k) - \nabla f_{s_k}(x^*)\|^2\right] + 2\mathbb{E}\left[\|\nabla f_{s_k}(y) - \nabla f(y) - \nabla f_{s_k}(x^*)\|^2\right] \\
&\leq 2L^2\mathbb{E}\left[\Delta_k^2\right] + 2\mathbb{E}\left[\|\nabla f_{s_k}(y) - \nabla f_{s_k}(x^*)\|^2\right] \\
&\leq 2L^2\mathbb{E}\left[\Delta_k^2\right] + 2L^2\mathbb{E}\left[\|y - x^*\|^2\right].
\end{aligned} \tag{40}$$

- 其中第一个不等式是因为$\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$，第二个不等式使用了有关二阶矩的不等式

$$\mathbb{E}[\|\xi - \mathbb{E}\xi\|^2] \leq \mathbb{E}[\|\xi\|^2].$$

SVRG算法

- 从(40)式看出，若x^k和y非常接近x^*，梯度估计的方差就很小.
- 频繁地更新y可以使得方差更小，但也增加了计算全梯度的次数.

SVRG算法的收敛性

下面给出SVRG算法的收敛性. 这里的收敛性是针对参考点序列$\{\tilde{x}^j\}$而言的.

定理 (SVRG算法的收敛性)

设m为利用每个$\tilde{x}^j$更新的次数. 设每个$f_i(x)$可微, 且梯度L-利普希茨连续; 函数$f(x)$强凸, 强凸参数为μ. 取步长$\alpha \in \left(0, \frac{1}{2L}\right]$, 并且$m$充分大使得

$$\rho = \frac{1}{\mu\alpha(1-2L\alpha)m} + \frac{2L\alpha}{1-2L\alpha} < 1, \tag{41}$$

则*SVRG*算法对于参考点$\tilde{x}^j$在函数值期望的意义下有Q-线性收敛速度:

$$\mathbb{E}f(\tilde{x}^j) - f(x^*) \leq \rho\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)]. \tag{42}$$

定理的证明

- 定义$\Delta_k = \|x^k - x^*\|$.
- 对于内层循环，

$$\begin{aligned}\mathbb{E}[\Delta_{k+1}^2] &= [\|x^{k+1} - x^*\|^2] = \mathbb{E}[\|x^k - \alpha v^k - x^*\|^2] \\ &= \mathbb{E}[\Delta_k^2] - 2\alpha\mathbb{E}[\langle v^k, x^k - x^*\rangle] + \alpha^2\mathbb{E}[\|v^k\|^2] \\ &= \mathbb{E}[\Delta_k^2] - 2\alpha\mathbb{E}[\langle \nabla f(x^k), x^k - x^*\rangle] + \alpha^2\mathbb{E}[\|v^k\|^2] \\ &\leq \mathbb{E}[\Delta_k^2] - 2\alpha\mathbb{E}[(f(x^k) - f(x^*))] + \alpha^2\mathbb{E}[\|v^k\|^2].\end{aligned}$$

- 构造辅助函数

$$\phi_i(x) = f_i(x) - f_i(x^*) - \nabla f_i(x^*)(x - x^*),$$

注意到$\phi_i(x)$也是凸函数且梯度L-利普希茨连续，因此有

$$\frac{1}{2L}\|\nabla\phi_i(x)\|^2 \leq \phi_i(x) - \phi_i(x^*)$$

- 展开$\phi_i(x)$与$\nabla\phi_i(x)$的表达式可得

$$\|\nabla f_i(x) - \nabla f_i(x^*)\|^2 \le 2L[f_i(x) - f_i(x^*) - \nabla f_i(x^*)^\top (x - x^*)].$$

- 对i从1到N进行求和，注意$\nabla f(x^*) = 0$:

$$\frac{1}{N}\sum_{i=1}^{N} \|\nabla f_i(x) - \nabla f_i(x^*)\|^2 \le 2L[f(x) - f(x^*)], \quad \forall x. \tag{43}$$

- 利用(40)式的推导过程可得

$$\mathbb{E}[\|v^k\|^2] \le 2\mathbb{E}[\|\nabla f_{s_k}(x^k) - \nabla f_{s_k}(x^*)\|^2] + 2\mathbb{E}[\|\nabla f_{s_k}(\tilde{x}^{j-1}) - \nabla f_{s_k}(x^*)\|^2].$$

对上式右侧第一项，有

$$\begin{aligned}
&\mathbb{E}[\|\nabla f_{s_k}(x^k) - \nabla f_{s_k}(x^*)\|^2] \\
=\ &\mathbb{E}[\mathbb{E}[\|\nabla f_{s_k}(x^k) - \nabla f_{s_k}(x^*)\|^2 | s_1, s_2, \cdots, s_{k-1}]] \\
=\ &\mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N} \|\nabla f_i(x^k) - \nabla f_i(x^*)\|^2\right] \le 2L\mathbb{E}[f(x^k) - f(x^*)],
\end{aligned}$$

- 类似地，对右侧第二项，有

$$\mathbb{E}[\|\nabla f_{s_k}(\tilde{x}^{j-1}) - \nabla f_{s_k}(x^*)\|^2] \leq 2L\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)].$$

- 最终可得对$\mathbb{E}[\|v^k\|^2]$的估计：

$$\mathbb{E}[\|v^k\|^2] \leq 4L(\mathbb{E}[f(x^k) - f(x^*)] + \mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)]).$$

- 将$\mathbb{E}[\|v^k\|^2]$的上界代入对$\mathbb{E}[\Delta_{k+1}^2]$的估计，就有

$$\begin{aligned}\mathbb{E}[\Delta_{k+1}^2] &\leq \mathbb{E}[\Delta_k^2] - 2\alpha\mathbb{E}[f(x^k) - f(x^*)] + \alpha^2\mathbb{E}[\|v^k\|^2] \\ &\leq \mathbb{E}[\Delta_k^2] - 2\alpha(1 - 2\alpha L)\mathbb{E}[f(x^k) - f(x^*)] \\ &\quad + 4L\alpha^2\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)].\end{aligned}$$

- 对k从1到m求和，并且注意到$x^1 = \tilde{x}^{j-1}$就可以得到

$$\begin{aligned}
&\mathbb{E}[\Delta_{m+1}^2] + 2\alpha(1-2\alpha L)\sum_{k=1}^{m}\mathbb{E}[f(x^k) - f(x^*)] \\
&\leq \mathbb{E}[\|\tilde{x}^{j-1} - x^*\|^2] + 4L\alpha^2 m\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)] \\
&\leq \frac{2}{\mu}\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)] + 4L\alpha^2 m\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)],
\end{aligned}$$

- 注意到$\tilde{x}^j = \frac{1}{m}\sum_{k=1}^{m} x^k$，所以

$$\begin{aligned}
&\mathbb{E}[f(\tilde{x}^j) - f(x^*)] \\
&\leq \frac{1}{m}\sum_{k=1}^{m}\mathbb{E}[f(x^k) - f(x^*)] \\
&\leq \frac{1}{2\alpha(1-2\alpha L)m}\left(\frac{2}{\mu} + 4mL\alpha^2\right)\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)] \\
&= \rho\mathbb{E}[f(\tilde{x}^{j-1}) - f(x^*)].
\end{aligned}$$