Lecture 20 - variance reduction

Tuesday, April 2, 2024 9:35 AM

today: • variance reduction perspective
• application to CRF

Variance reduction idea

X & Y are R.V.'s

goal: estimate $\mathbb{E}X$ using M.C. samples

Suppose: $\mathbb{E}Y$ is cheap to compute and Y is correlated with X

consider estimator "control variate" technique

$\alpha \in [0,1]$ $\quad \Theta_\alpha \triangleq \alpha(X-Y) + \mathbb{E}Y$ to approximate $\mathbb{E}X$

α: convex combo. coeff. between $\mathbb{E}X$ & $\mathbb{E}Y$

properties: $\mathbb{E}\Theta_\alpha = \alpha\mathbb{E}X + (1-\alpha)\mathbb{E}Y$ $\Rightarrow$ unbiased (ie. $\mathbb{E}\Theta_\alpha = \mathbb{E}X$)

if $\alpha = 1$ or $\mathbb{E}X = \mathbb{E}Y$ [not interesting]

variance: $\mathrm{Var}(\Theta_\alpha) = \alpha^2[\mathrm{var}X + \mathrm{var}Y - 2\mathrm{cov}(X,Y)]$

variance reduction?

for $\alpha = 1$ (unbiased setting) $\Theta_\alpha = X + (\mathbb{E}Y - Y)$

correction

SGD setting

X is $\nabla f_i(x_t)$; $\mathbb{E}X$ = batch gradient

SAG/SAGA algorithm: Y is g_i [past stored gradient]

$$\mathbb{E}Y = \frac{1}{n}\sum_{i=1}^{n} g_i$$

SAG alg.: $\alpha = \frac{1}{n}$ (biased)

SAGA alg.: $\alpha = 1$ $\Rightarrow$ (unbiased)

$\alpha(X-Y)$ + $\mathbb{E}Y$

SAG: $x_{t+1} = x_t - \gamma\left[\frac{1}{n}(\nabla f_{i_t}(x_t) - g_{i_t}^{(t)}) + \frac{1}{n}\sum_j g_j^{(t)}\right]$ (biased)

SAGA: $x_{t+1} = x_t - \gamma\left[1(\nabla f_{i_t}(x_t) - g_{i_t}^{(t)}) + \frac{1}{n}\sum_j g_j^{(t)}\right]$ (unbiased)

SVRG: $x_{t+1} = x_t - \gamma\left[(\nabla f_{i_t}(x_t) - \nabla f_{i_t}(x_{old})) + \frac{1}{n}\sum \nabla f_i(x_{old})\right]$ (unbiased)

SAGA: $x_{t+1} = x_t - \gamma \left[\nabla f_{i_t}(x_t) - y_{i_t} + \frac{1}{n} \sum_j y_j \right]$ (unbiased)

SVRG: (stochastic variance reduced gradient) $x_{t+1} = x_t - \gamma \left[\left(\nabla f_{i_t}(x_t) - \nabla f_{i_t}(x_{old}) \right) + \frac{1}{n} \sum_j \nabla f_j(x_{old}) \right]$ (unbiased)

↳ x_{old} is updated from outer loop (infrequently)

SVRG alg.:

for $k = 0, \dots,$ (outer loop)

compute $g_{ref} \triangleq \frac{1}{n} \sum_j \nabla f_j(x^{(k)})$

for $t = 0, \dots, T_{max}$

sample i_t

$x_{t+1}^{(k)} = x_t^{(k)} - \gamma \left[\nabla f_{i_t}(x_t^{(k)}) - \nabla f_i(x^{(k)}) + g_{ref} \right]$

end

$x^{(k+1)} = x_{T_{max}}^{(k)}$

Question is:
- what is T_{max}?
- what is γ?

original SVRG convergence result need $\gamma \lesssim \frac{1}{L}$

$T_{max} \geq \approx \frac{L}{\mu} = \kappa$ → to run alg., need to know κ

⇒ not adaptive to local strong convexity

tweak of SVRG (now called "Loopless")

[Hoffman & al. NeurIPS 2015] $T_{max} \sim Geom(\dots)$

[at inner loop, do a batch gradient computation with prob. $\frac{1}{n}$]

then, get same convergence result as SAGA

↳ comp. cost.: size of inner loop $\mathbb{E}[T_{max}] = n$

overall cost of SVRG $\approx$ 3 (SGD cost) for n updates

note: interpolation regime: $\|\nabla f_i(x^*)\| = 0 \ \forall i$

vs. yes! $\| \frac{1}{n} \sum_i \nabla f_i(x^*) \| = 0$

↳ get similar rate as SVRG/SAGA with SGD with constant step

↳ get similar rate as SVRG/SAGA with SGD with constant step

CRF optimization

SVM struct

primal: $\min_w \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^n H_i(w)$, where $H_i(w) = \max_{\tilde{y}} \ell_i(\tilde{y}) - w^T\psi_i(\tilde{y})$ (hinge)

dual: $\max_{\alpha_i \in \Delta_{|\mathcal{Y}_i|}} -\frac{\lambda}{2}\|w(\alpha)\|^2 + \frac{1}{n}\sum_{i=1}^n \ell_i^T \alpha_i$

CRF

$\min_w \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^n -\log p(y^{(i)} | x^{(i)}, w)$, where $-\log p(y^{(i)} | x^{(i)}, w) = \log\left(\sum_{\tilde{y}} \exp(-w^T\psi_i(\tilde{y}))\right)$

$\max_{\alpha_i \in \Delta_{|\mathcal{Y}_i|}} -\frac{\lambda}{2}\|w(\alpha)\|^2 + \frac{1}{n}\sum_{i=1}^n H_i(\alpha_i)$ (entropy), where $H_i(\alpha_i) \triangleq -\sum_{\tilde{y}} \alpha_i(\tilde{y}) \log \alpha_i(\tilde{y})$

KKT $\rightarrow$ $w(\alpha) = \frac{1}{\lambda n}\sum_i \sum_{\tilde{y}} \alpha_i(\tilde{y})\psi_i(\tilde{y})$

$= \frac{1}{\lambda n}\sum_i \sum_{c \in C} \sum_{\tilde{y}_c} \mu_{i,c}(\tilde{y}_c)\psi_{i,c}(\tilde{y}_c)$ ← from MRF

$p(y|x;w) \propto \exp(\langle w, \varphi(x,y)\rangle)$

(*) at optimality

$\alpha_i^*(y) = p(y | x^{(i)}; w(\alpha^*))$

$\Downarrow$

$\alpha_i^* \in$ interior of $\Delta_{|\mathcal{Y}_i|}$ (fully dense)

unlike sparse soln for SVM struct

10h20

CRF optimization

- primal is smooth [vs. non-smooth for SVM struct]
  - for a while, batch L-BFGS was method of choice [batch => slow for large n]
  - [Collins & al. JMLR 2008]: online exponentiated gradient (OEG)

block-coordinate method on dual; exponentiated gradient step on block

$\alpha_i(\tilde{y})^{(t+1)} \propto \alpha_i(\tilde{y})^{(t)} \exp(-\gamma_t \nabla_{\alpha_i} D(\alpha^{(t)}))$ ← dual objective

EG alg → proximal gradient step using $KL(\alpha \| \alpha^{(t)})$ as Bregman divergence for prox term.

get linear convergence rate with cheap $O(1)$ updates (like SGD) [vs. $O(n)$ for batch methods]

[can think of it as a variance reduced method as well?]

- SAGA for CRF [Schmidt & al. AISTATS 2015]

$w^{(t+1)} = (1-\lambda\gamma_t)w^{(t)} - \gamma_t\left[\nabla f_{i_t}(w_t) - g_i^{(t)} + \frac{1}{n}\sum_j g_j^{(t)}\right]$

- SDCA (stochastic dual coordinate ascent) & [illegible]

$\alpha_i^{(t+1)}(\tilde{y}) = (1-\gamma_t)\alpha_i^{(t)}(\tilde{y}) + \gamma_t \tilde{s}_i^{(t)}(\tilde{y})$

$\tilde{s}_i^{(t)}(\tilde{y}) \in [0,1]$ stabilize — related to subgradient

[stochastic dual coordinate ascent]

↳ SOTA for ERF [Le Priol & al. UAI 2016] thanks to efficient line search

as relaxed fixed point update

$\alpha_i^{t+1} = \rho(y|x_i; w(\alpha^t))\ \forall i$

$\rho(y|x_i; w(\alpha^{t+1})) \triangleq \alpha_i(w)$

↳ $[\alpha_i]$ stabilize

related to subgradient in primal

[note: BCFW is a special case of SDCA on SVM dual obj.]

proximal gradient method

↳ generalization of projected gradient method to other non-smooth fct.

composite framework $F(w) \triangleq f(w) + \Omega(w)$ where f is convex & L-smooth

Ω " " but not nec. smooth

- constrained opt.: $\Omega(w) = \delta_M(w) \triangleq \begin{cases} 0 & \text{if } w \in M \\ +\infty & \text{o.w.} \end{cases}$ "indicator fct." on M
- ℓ_1-regularization $\Omega(w) = \|w\|_1$

proximal gradient update:

$$w_{t+1} = \underset{w}{\operatorname{argmin}} \underbrace{f(w_t) + \langle \nabla f(w_t), w - w_t \rangle + \frac{1}{2\gamma_t}\|w - w_t\|_2^2}_{B_t(w)} + \Omega(w)$$

- if $\gamma \leq \frac{1}{L}$, then $f(w) \leq B_t(w)\ \forall w$
- we can rewrite $B_t(w) = \frac{1}{2\gamma_t}\|w - [w_t - \gamma_t \nabla f(w_t)]\|_2^2 + \text{cst.}$ (by completing the square)

$\Rightarrow$ if $\Omega(w) = \delta_M(w)$: we get projected grad. alg.

$$w_{t+1} = \operatorname{Prox}_{\gamma_t}^{\Omega}(w_t - \gamma_t \nabla f(x_t))$$

↳ "proximal operator"

$$\operatorname{prox}_\gamma^\Omega(z) \triangleq \underset{w}{\operatorname{argmin}} \left\{ \Omega(w) + \frac{1}{2\gamma}\|w - z\|^2 \right\}$$

can be replaced by Bregman divergence to get other generalizations (e.g. CEG)

⊛ like projection, prox operator is non-expansive (i.e. 1-Lipschitz) (recall lecture 1/ landscape of risks)

i.e. $\|\operatorname{prox}_\gamma^\Omega(w) - \operatorname{prox}_\gamma^\Omega(w')\|_2 \leq \|w - w'\|_2$

i.e. $\|\text{prox}_\gamma^\Omega(w) - \text{prox}_\gamma^\Omega(w')\|_2 \leq \|w - w'\|_2$ (nice)

$\Rightarrow$ convergence rate for prox grad. method on $F = f + \Omega$ are same as unconstrained gradient descent on f

* to be useful, need $\text{prox}_\gamma^\Omega$ to be efficiently computable

$\text{prox}_\gamma^{\|\cdot\|_1}(z) = \arg\min_w \|w\|_2 + \frac{1}{2\gamma}\|w - z\|_2^2$

"soft-thresholding" (component-wise) $= \begin{cases} \text{sgn}(z_d)[z_d - \gamma] & \text{if } |z_d| \geq \gamma \\ 0 & \text{o.w.} \end{cases}$

used eg. for lasso : ℓ_1-regularized least-squares

FISTA $\rightarrow$ accelerated prox gradient method

$\hookrightarrow$ SOTA for batch lasso

* scikit-learn $\rightarrow$ use (prox) SAGA for lasso & ℓ_1-reg. log regression

prox SAGA $w_{t+1} = \text{Prox}_\gamma^\Omega\left(w_t - \gamma\left[\nabla f_{i_t}(w_t) - g_{i_t}^{(t)} + \frac{1}{n}\sum_j g_j^{(t)}\right]\right)$

could accelerate prox SAGA using "catalyst" (see next class)