Lecture 17 - scribbles - FW for SVMstruct

Tuesday, March 20, 2018 14:27

today: • FW for SVMstruct
• BCFW
• variance reduced SGD [SAG]

FW for SVMstruct:

FW on dual of SVMstruct objective:

$$\min_{\alpha_i \in \Delta_{|\mathcal{Y}_i|}} \underbrace{\frac{\lambda \|A\alpha\|^2}{2} - b^T\alpha}_{f(\alpha) \triangleq -d(\alpha)}$$

$$M = \bigoplus_{i=1}^{n} \Delta_{|\mathcal{Y}_i|}$$

$$(\nabla f(\alpha))_{i,y} = -\frac{1}{n} H(y; w(\alpha)) \qquad w(\alpha) = A\alpha$$

$$= (\lambda A^T A\alpha - b)_{i,y} \qquad b_i(y) = \frac{1}{n} \ell_i(y)$$

$$A_{i,y} = \frac{\psi_i(y)}{\lambda n}$$

FW step: $\min_{s \in M} \langle s, \nabla f(\alpha_t) \rangle$

$$= \sum_{i=1}^{n} \min_{s_i \in \Delta_{|\mathcal{Y}_i|}} \langle s_i, \nabla_i f(\alpha_t) \rangle$$

$\alpha^{(t)} \xrightarrow{A} w^{(t)}$

$s_i^{(t)} = \delta_{\hat{y}_i^{(t)}}$ where $\hat{y}_i^{(t)} = \operatorname{argmax}_{\tilde{y} \in \mathcal{Y}_i} H_i(\tilde{y}; w^{(t)})$ ← loss-augmented decoding

$$\alpha^{(t+1)} = (1-\gamma)\alpha^{(t)} + \gamma s^{(t)}$$

A

$$w^{(t+1)} = (1-\gamma)\underbrace{A\alpha^{(t)}}_{w^{(t)}} + \gamma \underbrace{As^{(t)}}_{\frac{1}{\lambda n}\sum_{i=1}^{n} \psi_i(\hat{y}_i^{(t)})}$$

recall primal:

$$p(w) = \frac{\lambda \|w\|^2}{2} + \frac{1}{n}\sum_{i=1}^{n} H_i(w)$$

$$H_i(w) = \max_{\tilde{y}} \ell_i(\tilde{y}) - w^T\psi_i(\tilde{y})$$

$$\nabla p(w) = \lambda w^{(t)} - \frac{1}{n}\sum_{i=1}^{n} \psi_i(\hat{y}_i^{(t)})$$

★ advantage of FW: adaptive step-size using analytic line search

$$\frac{\partial P}{\partial} = \lambda w^{(t)} - \frac{1}{n}\sum_{i=1}^{n} \psi_i(\hat{y}_i^{(t)})$$

using analytic line search

$$\gamma_t^* = \underset{\gamma \in [0,1]}{\operatorname{argmin}} \; f(\alpha^{(t)} + \gamma(s^{(t)} - \alpha^{(t)}))$$

$$w^{(t+1)} = w^{(t)} - \beta \frac{\partial P}{\partial w} = (1 - \lambda\beta) w^{(t)} + \frac{\beta}{n}\sum_{i=1}^{n} \psi_i(\hat{y}_i^{(t)})$$

if set $\boxed{\beta = \frac{\gamma}{\lambda}}$; then batch subgradient step on primal is equivalent to batch FW step on dual

recall: subgradient method convergence result $O(\frac{1}{t})$

needed a step size $\beta_t \leq \frac{1}{\mu} O(\frac{1}{t})$

FW method with fixed step-size schedule uses $\gamma_t = \frac{2}{t+2}$

gives $d(\alpha^*) - d(\alpha^{(t)}) \leq \frac{2C_f}{t+2}$

here, FW-gap $\langle -\nabla f(\alpha^{(t)}), s^{(t)} - \alpha^{(t)} \rangle = \frac{1}{n}\sum_{i=1}^{n}\left[H_i(\hat{y}_i^{(t)}; w^{(t)}) - \sum_{\tilde{y} \in Y_i} \alpha_i^{(t)}(\tilde{y})\, H_i(\tilde{y}; w^{(t)}) \right]$

$= P(w(\alpha^{(t)})) - d(\alpha^{(t)})$

(Lagrangian gap of lecture 14?)

[note: FW gap is not always such Lagrangian gap]
e.g. CRF objective

* note one can show
  - that $\min_{s \leq t} g_s \leq 3 \cdot \frac{2C_f}{t+2}$ (FW gap)
  - also have $g^{FW}(\hat{x}_{WA}^{(t)}) \leq 3 \cdot \frac{2C_f}{t+2}$

($\hat{x}_{WA}^{(t)}$: weighted average)

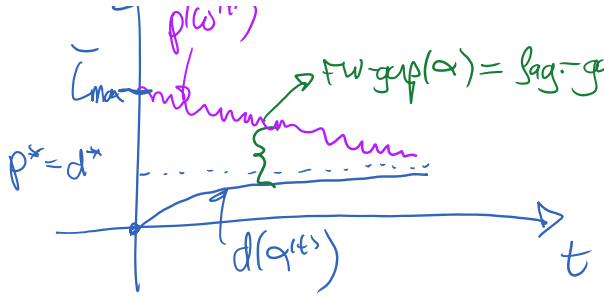
when $g(\cdot)$ is convex [this is case for f being quadratic]

$$P(0) = \frac{1}{n}\sum_{i=1}^{n} \max_{\tilde{y}} l_i(\tilde{y}) = \tilde{L}_{max}$$

FW-gap(α) = lag.-gap

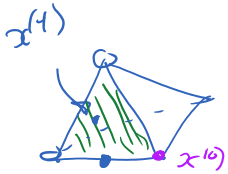
$p(0) = \frac{1}{n}\sum_{i=1}^{n} \max_{\tilde{y}} \ell_i(\tilde{y}) = L_{max}$

$d(\alpha^{(0)}) = 0$



FCFW "fully-corrective FW" variant.

algorithm: re-optimize f over conv-hull $\left(\{s^{(u)}\}_{u=0}^{t}\right)$



[special case: min norm point (MNP) algorithm
→ state of art for submodular optimization]

turns out that

> (batch) FCFW on dual of svmstruct
> is equivalent to the constraint generation / cutting plane approach for 1-slack formulation

why? every $s^{(t)}$ corresponds $\{\hat{y}_i^{(t)}\}_{i=1}^{n}$

$\alpha \in \text{conv}(\{s^{(u)}\}_{u \le t})$ $\quad \alpha = \sum_{u \le t} \tilde{\alpha}_u s^{(u)}$

$$w = A\alpha = \sum_{u \le t} \tilde{\alpha}_u \frac{1}{\lambda n}\left(\sum_{i=1}^{n} \psi_i(\hat{y}_i^{(u)})\right)$$

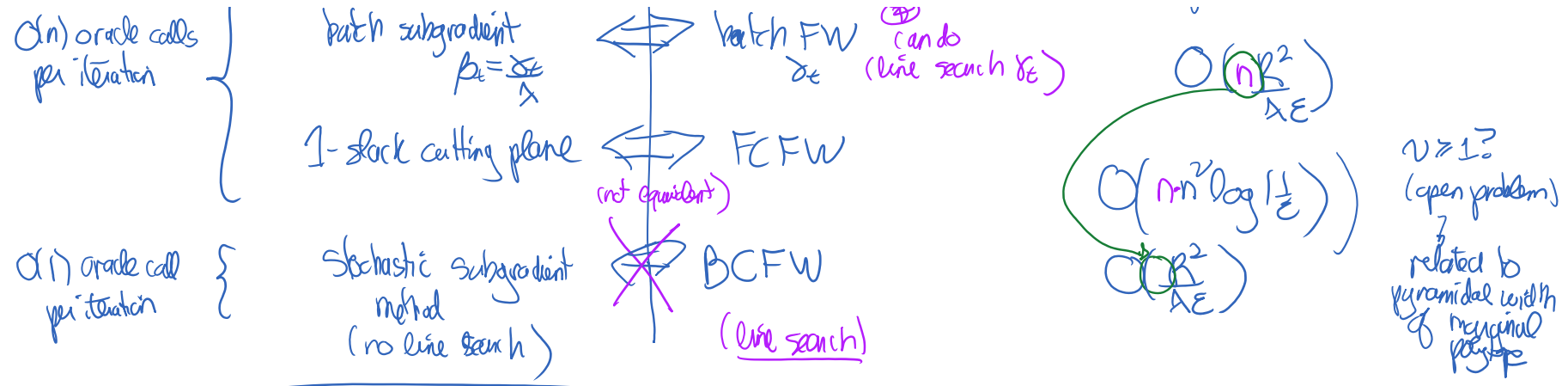
to summarize

$O(n)$ oracle calls per iteration

primal problem		dual problem
batch subgradient $\beta_t = \frac{\partial f}{\partial \alpha}$	⟺	batch FW γ_t (can do line search γ_t)

complexity in # of oracle calls

$O(nk^2)$



small QP formulation
when $H_i(w)$ = tractable LP ← cost per iteration $O((n \cdot \#\text{parts} + d)^3)$
# of iterations $O(\square \log\frac{1}{\varepsilon})$

Block-coordinate optimization

[wright reference 2016 survey?] →

"huge scale" optimization → [Nesterov 2010-2012]

proposed: randomized block-coordinate projected gradient method

setup: $\min f(x)$ s.t. $x \in \bigtimes_{i=1}^{n} M_i$

$x = (\underbrace{x_1}_{\text{block}}, \dots, x_n)$

$D = \begin{pmatrix} \vdots \\ D_i \\ \vdots \end{pmatrix}$

alg: pick i at random

then let $\begin{cases} x_i^{(t+1)} = \text{Proj}_{M_i}\left(x_i^{(t)} - \frac{1}{L_i} D_i f(x^{(t)})\right) \\ x_j^{(t+1)} = x_j^{(t)} \quad \forall j \neq i \end{cases}$ ← Lipschitz constant of $D_i f(x)$

[only update block i at iteration t]

Nesterov showed: $\mathbb{E} f(x^{(t)}) - f(x^*) \leq \frac{2}{t+4}\left[\frac{1}{n}\sum_{i=1}^{n} L_i\right]\|x_0 - x^*\|^2$ (for convex f)

Block-coordinate FW (BCFW): idea: do FW step on block i

alg: for $t=0, \dots$

pick i at random

let $s_i^{(t)} \triangleq \underset{s_i \in M_i}{\operatorname{argmin}} \langle s_i, \nabla_i f(x^{(t)}) \rangle$ [FW corner for block i]

$S_{[i]} = \begin{pmatrix} 0 \\ \vdots \\ s_i \\ \vdots \\ 0 \end{pmatrix}$

$$\begin{cases} x_i^{(t+1)} = x_i^{(t)}(1-\gamma_t) + \gamma_t s_i^{(t)} \\ x_j^{(t+1)} = x_j^{(t)} \quad \text{for } j \neq i \end{cases}$$

$\gamma_t = \begin{cases} \text{line search: } \underset{\gamma \in [0,1]}{\operatorname{argmin}} f(x^{(t)} + \gamma(s_{[i]}^{(t)} - x_{[i]}^{(t)})) \\ \frac{2n}{t+2n} \end{cases}$ n = # of blocks

* an important property: $FW\text{-}gap = \max_{s \in M} \langle -\nabla f(x), s - x \rangle \overset{\text{product structure}}{=} \sum_i \underbrace{\max_{s_i \in M_i} \langle -\nabla f(x), s_i - x_i \rangle}_{g_i(x)}$ "block FW gap"

$$g^{FW}(x) = \sum_i g_i^{BCFW}(x)$$

↳ motivated "gap sampling" variant [Osokin et al. ICML 2016]

as before, can show $g_i(x) \geq f(x) - \min_{y: y_j = x_j \ \forall j \neq i} f(y)$

convergence result for BCFW: $C_f^{(i)} \leq L_i \operatorname{diam}(M_i)^2$ $\quad C_f^{\otimes} \triangleq \sum_{i=1}^n C_f^{(i)}$

$$\mathbb{E}[f(x^{(t)})] - f(x^*) \leq \frac{2n}{t+2n}\left[C_f^{\otimes} + f(x^{(0)}) - f(x^*)\right] \quad \text{for } \gamma_t = \frac{2n}{t+2n}$$

$\triangleq \varepsilon_t$

If you use line search: $\varepsilon_t \leq \frac{2n\, C_f^{\otimes}}{t - t_0 + 2n}$ for $t \geq t_0$

where $t_0 \leq n \log \frac{2(f(x^{(0)}) - f(x^*))}{C_f^{\otimes}}$ time to ensure $\varepsilon_{t_0} \leq C_f^{\otimes}$

one can show that $C_f^{\otimes} \leq C_f$ for quadratic functions

Compare batch FW $\longrightarrow \frac{2C_f}{t+2}$

vs.

BCFW with line search $\longrightarrow \frac{2n\, C_f^{\otimes}}{t - t_0 + 2n}$

ie. # of oracle calls to reach ε-error

but BCFW is n times cheaper than batch FW — FW: $O\left(\frac{nC_f}{\varepsilon}\right)$

$\Rightarrow$ BCFW is "never" slower than batch FW — BCFW: $O\left(\frac{nC_f^{\otimes}}{\varepsilon}\right)$

extensions: } ICML 2016

- non-uniform sampling eg using $\begin{cases} C_f^{(i)} \\ g_i(x) \end{cases}$
- using away-step, etc... to "get" linear convergence

Application of BCFW for SVMstruct:

- getting s_i is one loss-augmented decoding call for example i
- $w = A\alpha = \sum_i \underbrace{A_i \alpha_i}_{\triangleq w_i}$ → you update $\begin{cases} w_i^{(t+1)} = w_i^{(t)} + \gamma(w_S^{(t)} - w_i^{(t)}) \\ w_j^{(t+1)} = w_j^{(t)} \quad \forall j \neq i \end{cases}$

need to store these in memory

for SVMstruct, $C_f^{(i)} \leq \frac{4R^2}{\lambda n^2}$ $\Rightarrow C_f^{(S)} = \frac{4R^2}{\lambda n} \approx \frac{C_f}{n}$

vs.

$C_f \leq \frac{4R^2}{\lambda}$

uses affine invariance crucially

ie: BCFW is "n times" faster than batch FW for SVMstruct ?

$C_f \leq L_{\|\cdot\|} \cdot \mathrm{diam}_{\|\cdot\|}(M)^2$

if use ℓ_2: $\|\cdot\|_2$ get bad bound $\mathrm{diam}(M)^2 = 2n$

Lipschitz constant in ℓ_2-norm = largest e-value of Hessian

$$\lambda A^T A = \frac{1}{\lambda n^2}\left(\langle \psi_i(y), \psi_j(y') \rangle\right)_{(i,y),(j,y')}$$

say e.g. $\langle \psi_i(y), \psi_j(y') \rangle \approx 1$ for lots of output

→ get largest e-value scale with dimension of matrix

$(\mathbb{1}\mathbb{1}^T)\mathbb{1} = \mathbb{1} \cdot \dim$

↳ here, dim is exponential ⇒ useless bound?