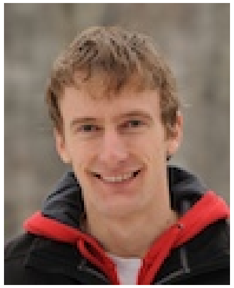
Block-Coordinate Frank-Wolfe for Structural SVMs

Martin Jaggi



Simon Lacoste-Julien

Mark Schmidt

Patrick Pletscher

CMAP ÉCOLE POLYTECHNIQUE

informatiques mathématiques

informatiques mathématiques

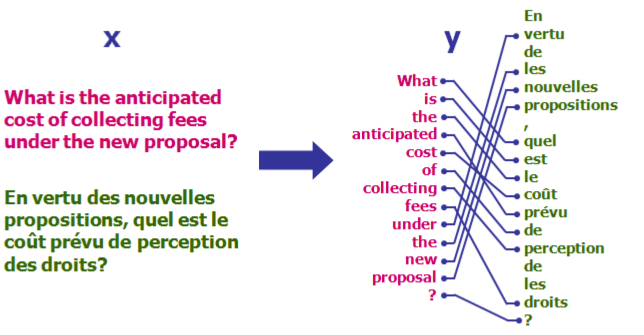
ETH

NIPS OPT2012 Workshop – Dec. 8th 2012

Structural SVM optimization

Structural SVM optimization

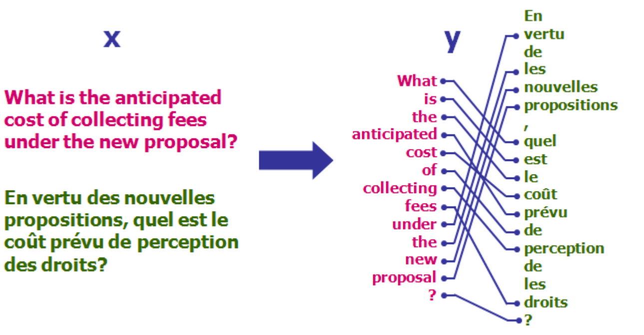
- structured prediction:



Structural SVM optimization

- structured prediction:
- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

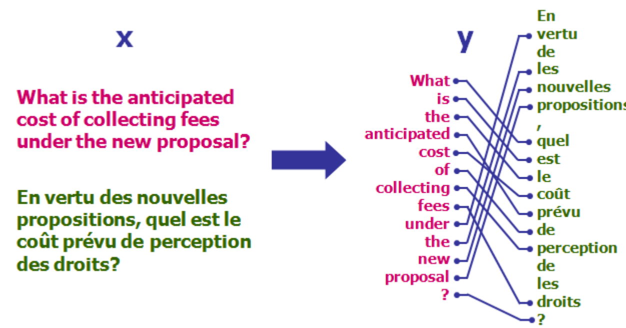


Structural SVM optimization

- structured prediction:
- learn linear classifier:

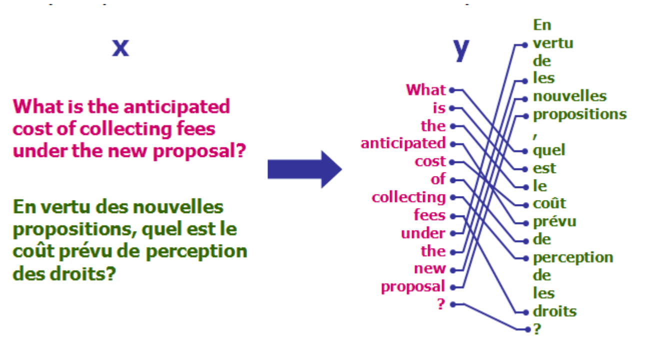
$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

← decoding



Structural SVM optimization

- structured prediction:



- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y} \in \mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

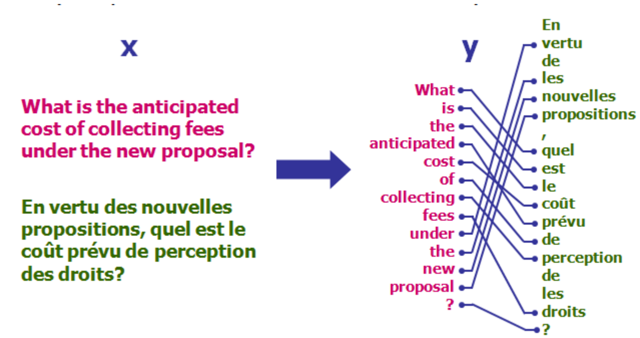
⟵ decoding

- structural SVM primal:

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2} \|\boldsymbol{w}\|^2 + \frac{1}{n} \sum_{i=1}^{n} \max_{\boldsymbol{y} \in \mathcal{Y}} \Big\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \big\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \big\rangle \Big\} - \big\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \big\rangle$$

Structural SVM optimization

- structured prediction:



- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

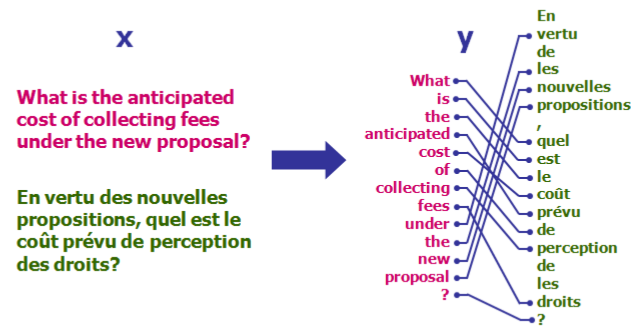
← decoding

- structural SVM primal:

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

vs. binary hinge loss: $\max\left\{0, 1 - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\, \boldsymbol{y}_i \right\rangle\right\}$

Structural SVM optimization



- structured prediction:
- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\mathrm{argmax}} \langle \boldsymbol{w}, \phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

← decoding

- structural SVM primal:

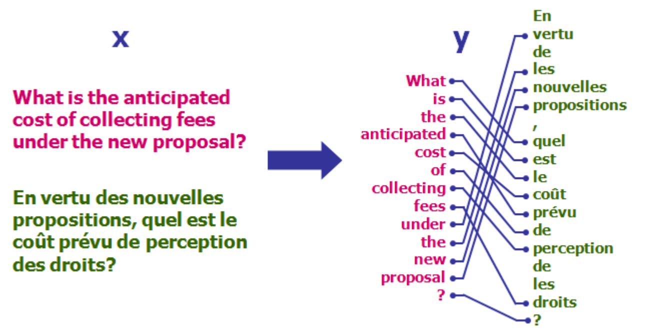
structured hinge loss:

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

vs. binary hinge loss: $\max \left\{ 0, 1 - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\, \boldsymbol{y}_i \right\rangle \right\}$

Structural SVM optimization

- structured prediction:



- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \phi(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

← decoding

- structural SVM primal:

structured hinge loss:

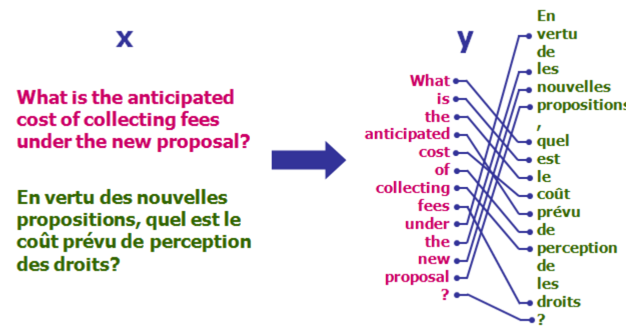
-> loss-augmented decoding

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

vs. binary hinge loss: $\max\left\{0, 1 - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i)\, \boldsymbol{y}_i \right\rangle\right\}$

Structural SVM optimization

- structured prediction:



- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

← decoding

- structural SVM primal:

structured hinge loss:

-> loss-augmented decoding

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2} \|\boldsymbol{w}\|^2 + \frac{1}{n} \sum_{i=1}^{n} \underbrace{\max_{\boldsymbol{y} \in \mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\}} - \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

vs. binary hinge loss: $\max \left\{ 0, 1 - \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i)\, \boldsymbol{y}_i \right\rangle \right\}$

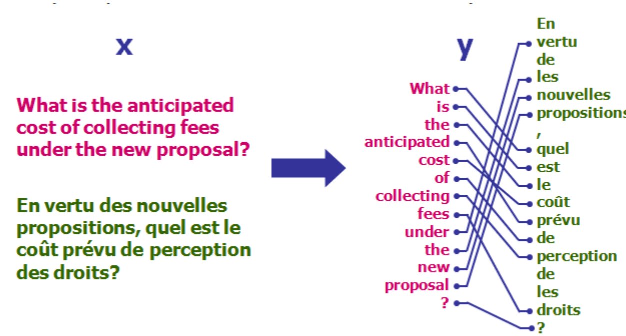
- structural SVM dual:

$$\max_{\boldsymbol{\alpha} \in \mathcal{M}} \quad \boldsymbol{b}^T \boldsymbol{\alpha} - \frac{\lambda}{2} \|A\boldsymbol{\alpha}\|^2$$

primal-dual pair: $\boldsymbol{w} = A\boldsymbol{\alpha}$

$$\mathcal{M} := \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$$

Structural SVM optimization



- structured prediction:
- learn linear classifier:

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmax}} \langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x},\boldsymbol{y})\rangle$$ ← decoding

- structural SVM primal:

structured hinge loss:

-> loss-augmented decoding

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i,\boldsymbol{y}) + \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y})\right\rangle \right\} - \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i,\boldsymbol{y}_i)\right\rangle$$

vs. binary hinge loss: $\max\left\{0, 1 - \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{x}_i)\, \boldsymbol{y}_i \right\rangle\right\}$

- structural SVM dual:

-> exp. number of variables!

$$\max_{\boldsymbol{\alpha}\in\mathcal{M}} \quad \boldsymbol{b}^T\boldsymbol{\alpha} - \frac{\lambda}{2}\|A\boldsymbol{\alpha}\|^2$$

primal-dual pair: $\boldsymbol{w} = A\boldsymbol{\alpha}$

$$\mathcal{M} := \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$$

Structural SVM optimization (2)

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

rate: after T passes through data:

$$\tilde{O}\left(\frac{1}{nT}\right)$$

$$O\left(\frac{1}{T}\right)$$

$$O\left(\frac{1}{nT}\right)$$

Structural SVM optimization (2)

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

- popular approaches:
  - stochastic subgradient descent
  - cutting plane method (SVMstruct)

rate: after T passes through data:

$$\tilde{O}\left(\frac{1}{nT}\right)$$

$$O\left(\frac{1}{T}\right)$$

$$O\left(\frac{1}{nT}\right)$$

Structural SVM optimization (2)

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

rate: after T passes through data:

- popular approaches:
  - stochastic subgradient descent — $\tilde{O}\left(\frac{1}{nT}\right)$
    - pros: online!
    - cons: sensitive to step-size; don't know when to stop
  - cutting plane method (SVMstruct) — $O\left(\frac{1}{T}\right)$

$O\left(\frac{1}{nT}\right)$

Structural SVM optimization (2)

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

- popular approaches:
  - stochastic subgradient descent
    - pros: online!
    - cons: sensitive to step-size; don't know when to stop
  - cutting plane method (SVMstruct)
    - pros: automatic step-size; duality gap
    - cons: batch! -> slow for large n

rate: after T passes through data:

$\tilde{O}\left(\frac{1}{nT}\right)$

$O\left(\frac{1}{T}\right)$

$O\left(\frac{1}{nT}\right)$

Structural SVM optimization (2)

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \rangle \right\} - \langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \rangle$$

rate: after T passes through data:

- popular approaches:
  - stochastic subgradient descent $\tilde{O}\left(\frac{1}{nT}\right)$
    - pros: online!
    - cons: sensitive to step-size; don't know when to stop
  - cutting plane method (SVMstruct) $O\left(\frac{1}{T}\right)$
    - pros: automatic step-size; duality gap
    - cons: batch! -> slow for large n
- **our approach**: block-coordinate Frank-Wolfe on dual

-> combines best of both worlds: $O\left(\frac{1}{nT}\right)$

Structural SVM optimization (2)

$$\min_{\boldsymbol{w}} \quad \frac{\lambda}{2}\|\boldsymbol{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \max_{\boldsymbol{y}\in\mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\} - \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) \right\rangle$$

rate: after T passes through data:

- popular approaches:
  - stochastic subgradient descent — $\tilde{O}\left(\frac{1}{nT}\right)$
    - pros: online!
    - cons: sensitive to step-size; don't know when to stop
  - cutting plane method (SVMstruct) — $O\left(\frac{1}{T}\right)$
    - pros: automatic step-size; duality gap
    - cons: batch! -> slow for large n
- **our approach**: block-coordinate Frank-Wolfe on dual

-> combines best of both worlds: $O\left(\frac{1}{nT}\right)$

  - **online!**
  - automatic step-size via analytic line search
  - duality gap
  - rates also hold for approximate oracles

Frank-Wolfe algorithm [Frank, Wolfe 1956]

(aka conditional gradient)

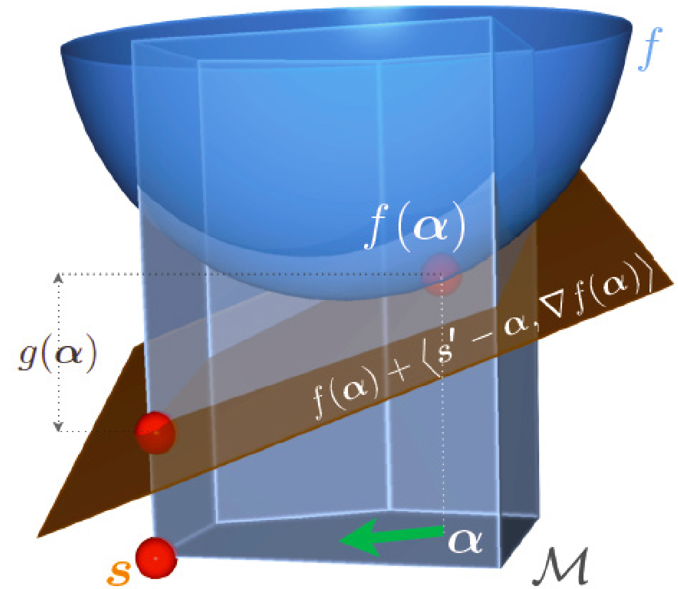
- alg. for constrained opt.: $\min_{\alpha \in \mathcal{M}} f(\alpha)$

  where:

  f convex & cts. differentiable

  $\mathcal{M}$ convex & compact

- FW algorithm – repeat:



Frank-Wolfe algorithm [Frank, Wolfe 1956]

(aka conditional gradient)

- alg. for constrained opt.: $\min_{\alpha \in \mathcal{M}} f(\alpha)$

  where:

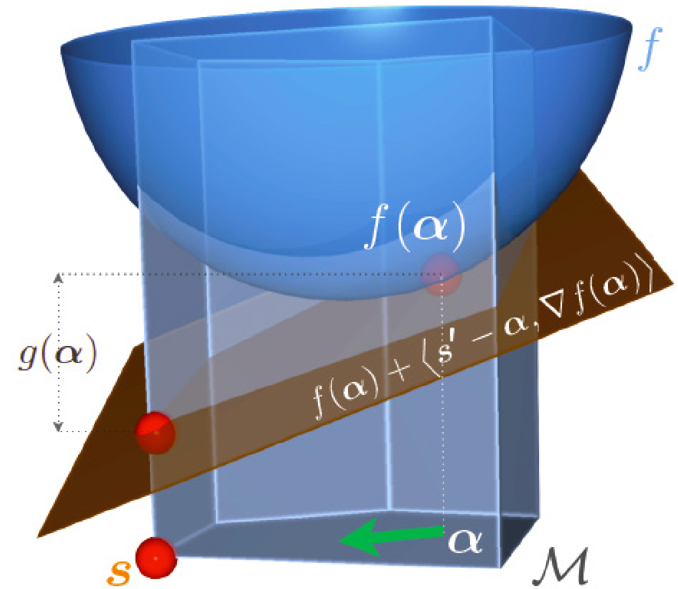
  f convex & cts. differentiable

  $\mathcal{M}$ convex & compact

- FW algorithm – repeat:

1) Find good feasible direction by minimizing linearization of f :

$$s_{t+1} \in \arg\min_{s' \in \mathcal{M}} \langle s', \nabla f(\alpha_t) \rangle$$



Frank-Wolfe algorithm [Frank, Wolfe 1956]

(aka conditional gradient)

- alg. for constrained opt.: $\min_{\alpha \in \mathcal{M}} f(\alpha)$

  where:

  f convex & cts. differentiable

  $\mathcal{M}$ convex & compact

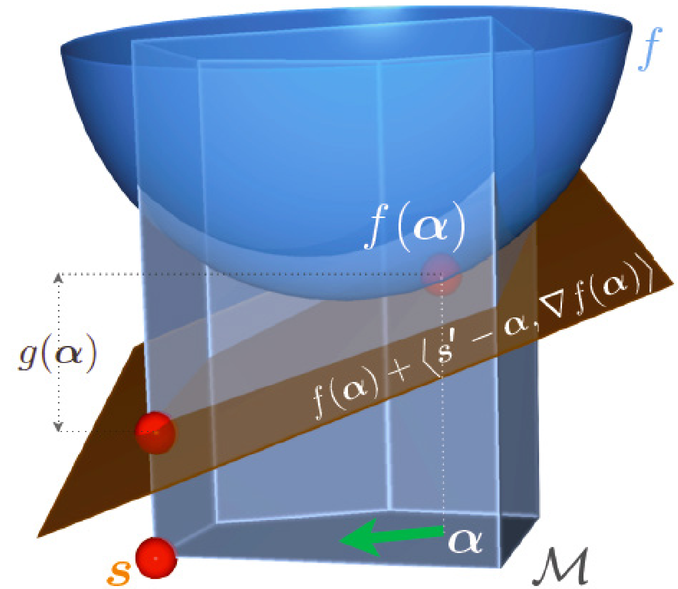
- FW algorithm – repeat:

1) Find good feasible direction by minimizing linearization of f :

$$s_{t+1} \in \arg\min_{s' \in \mathcal{M}} \langle s', \nabla f(\alpha_t) \rangle$$

2) Take convex step in direction:

$$\alpha_{t+1} = (1 - \gamma_t)\, \alpha_t + \gamma_t\, s_{t+1}$$



Frank-Wolfe algorithm [Frank, Wolfe 1956]

(aka conditional gradient)

- alg. for constrained opt.: $\min_{\alpha \in \mathcal{M}} f(\alpha)$

  where:

  f convex & cts. differentiable

  $\mathcal{M}$ convex & compact

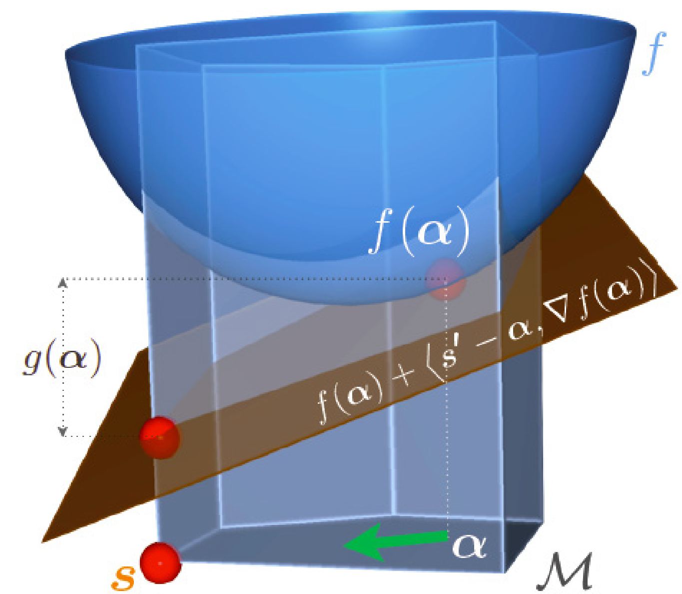
- FW algorithm – repeat:

1) Find good feasible direction by minimizing linearization of f :

$$s_{t+1} \in \arg\min_{s' \in \mathcal{M}} \langle s', \nabla f(\alpha_t) \rangle$$

2) Take convex step in direction:

$$\alpha_{t+1} = (1 - \gamma_t)\, \alpha_t + \gamma_t\, s_{t+1}$$



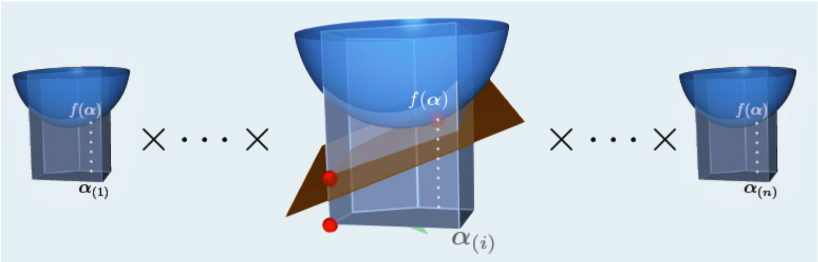
- Properties: O(1/T) rate
  - sparse iterates
  - get duality gap $g(\alpha)$ for free
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$



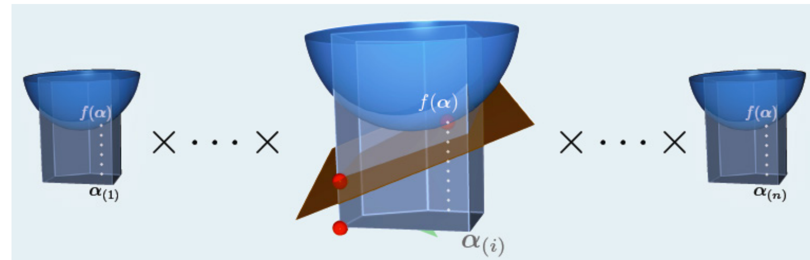
- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$



- pick i at random; update only block i with a FW step:

$$\boldsymbol{s}_{(i)} = \operatorname*{argmin}_{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle$$

$$\boldsymbol{\alpha}_{(i)}^{(k+1)} = (1-\gamma)\boldsymbol{\alpha}_{(i)}^{(k)} + \gamma \boldsymbol{s}_{(i)}$$

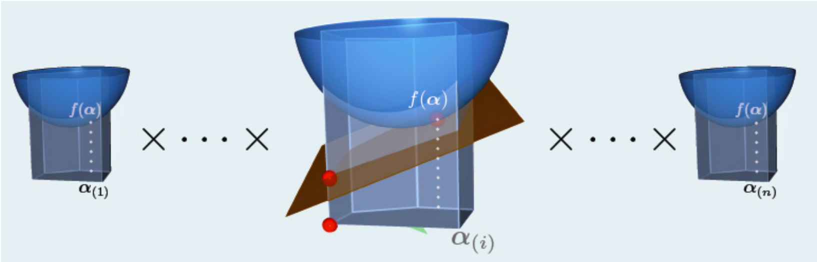
- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$



- pick i at random; update only block i with a FW step:

$$\boldsymbol{s}_{(i)} = \operatorname*{argmin}_{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle$$

$$\boldsymbol{\alpha}_{(i)}^{(k+1)} = (1-\gamma)\boldsymbol{\alpha}_{(i)}^{(k)} + \gamma \boldsymbol{s}_{(i)}$$

- we proved **same** O(1/T) rate

as batch FW

-> each step **n times cheaper** though

-> constant can be the same (SVM e.g.)

- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$

- pick i at random; update only block i with a FW step:

$$\boldsymbol{s}_{(i)} = \underset{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}}{\text{argmin}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle$$

$$\boldsymbol{\alpha}_{(i)}^{(k+1)} = (1-\gamma)\boldsymbol{\alpha}_{(i)}^{(k)} + \gamma \boldsymbol{s}_{(i)}$$

- we proved **same** O(1/T) rate

as batch FW

-> each step **n times cheaper** though

-> constant can be the same (SVM e.g.)

- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$

structural SVM:

$$\max_{\boldsymbol{\alpha} \in \mathcal{M}} \quad \boldsymbol{b}^T \boldsymbol{\alpha} - \frac{\lambda}{2} \|A\boldsymbol{\alpha}\|^2$$

$$\mathcal{M} := \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$$

- pick i at random; update only block i with a FW step:

$$\boldsymbol{s}_{(i)} = \operatorname*{argmin}_{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle$$

$$\boldsymbol{\alpha}_{(i)}^{(k+1)} = (1-\gamma)\boldsymbol{\alpha}_{(i)}^{(k)} + \gamma \boldsymbol{s}_{(i)}$$

- we proved **same** O(1/T) rate

as batch FW

-> each step **n times cheaper** though

-> constant can be the same (SVM e.g.)

- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$

structural SVM:

$$\max_{\boldsymbol{\alpha} \in \mathcal{M}} \quad \boldsymbol{b}^T \boldsymbol{\alpha} - \frac{\lambda}{2} \|A\boldsymbol{\alpha}\|^2$$

$$\mathcal{M} := \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$$

- pick i at random; update only block i with a FW step:

$$\boldsymbol{s}_{(i)} = \underset{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}}{\text{argmin}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle$$

$$\boldsymbol{\alpha}_{(i)}^{(k+1)} = (1-\gamma)\boldsymbol{\alpha}_{(i)}^{(k)} + \gamma \boldsymbol{s}_{(i)}$$

key insight:

- we proved **same** O(1/T) rate

as batch FW

-> each step **n times cheaper** though

-> constant can be the same (SVM e.g.)

- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Block-Coordinate Frank-Wolfe (new!)

- for constrained optimization over compact **product domain**:

$$\min_{\boldsymbol{\alpha} \in \mathcal{M}^{(1)} \times \ldots \times \mathcal{M}^{(n)}} f(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = (\boldsymbol{\alpha}_{(1)}, \ldots, \boldsymbol{\alpha}_{(n)})$$

structural SVM:

$$\max_{\boldsymbol{\alpha} \in \mathcal{M}} \quad \boldsymbol{b}^T \boldsymbol{\alpha} - \frac{\lambda}{2} \|A\boldsymbol{\alpha}\|^2$$

$$\mathcal{M} := \Delta_{|\mathcal{Y}_1|} \times \ldots \times \Delta_{|\mathcal{Y}_n|}$$

- pick i at random; update only block i with a FW step:

$$\boldsymbol{s}_{(i)} = \underset{\boldsymbol{s}'_{(i)} \in \mathcal{M}^{(i)}}{\text{argmin}} \left\langle \boldsymbol{s}'_{(i)}, \nabla_{(i)} f(\boldsymbol{\alpha}^{(k)}) \right\rangle \Leftrightarrow$$

key insight:

$$\max_{\boldsymbol{y} \in \mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) + \left\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}) \right\rangle \right\}$$

loss-augmented decoding

$$\boldsymbol{\alpha}_{(i)}^{(k+1)} = (1-\gamma)\boldsymbol{\alpha}_{(i)}^{(k)} + \gamma \boldsymbol{s}_{(i)}$$

- we proved **same** O(1/T) rate

as batch FW

-> each step **n times cheaper** though

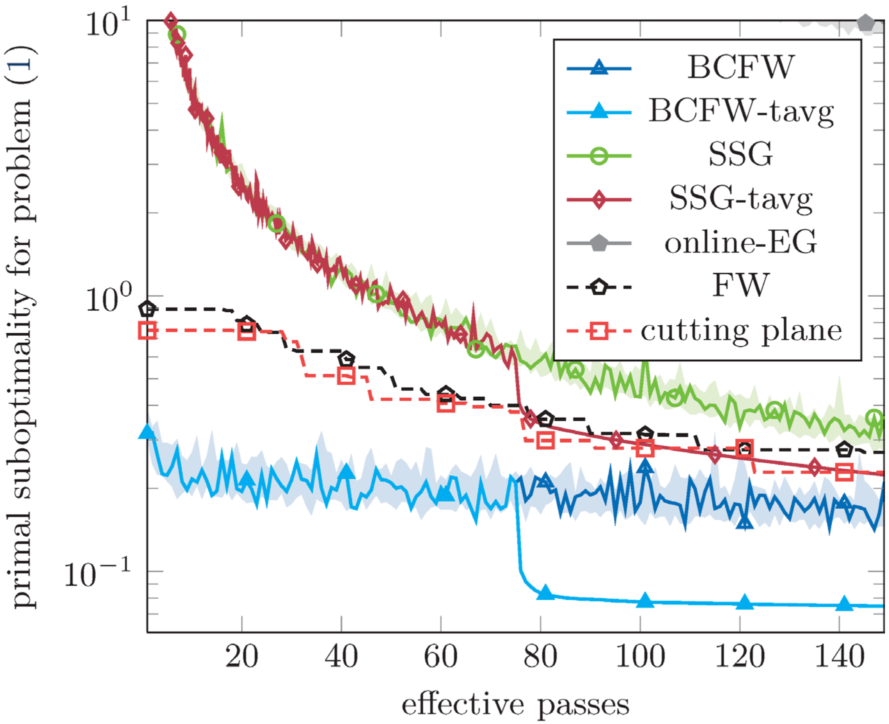
-> constant can be the same (SVM e.g.)

- Properties: O(1/T) rate
  - sparse iterates
  - duality gap guarantees
  - rate holds even if linear subproblem solved **approximately**

Experiments

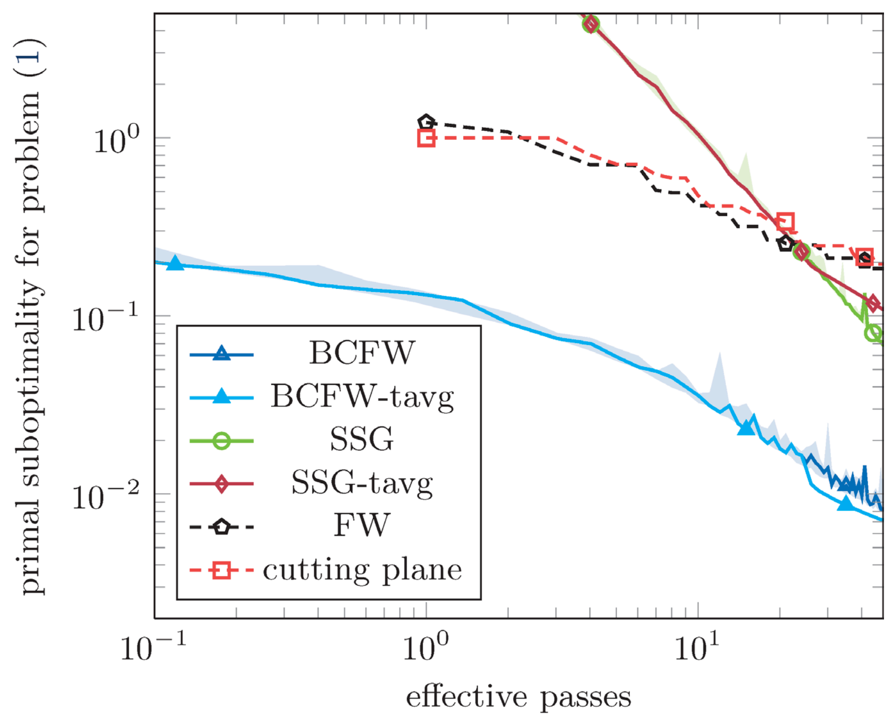
$\lambda = 1/n$

OCR dataset



$n =$ 6k, $d =$ 4k

CoNLL dataset



$n =$ 9k, $d =$ 1.6M

Conclusion

- new block-coordinate variant of Frank-Wolfe algorithm
  - same convergence rate but with cheaper iteration cost
- applied to structural SVM, yields:
  - online algorithm
  - optimal step-size computed in close form
  - duality gap
  - rates hold with approximate oracles