機械学習における連続最適化の新しいトレンド

冨岡 亮太 [1]

共同研究者: 鈴木 大慈 [1] **杉山 将** [2]

[1] **東京大学**
[2] **東京工業大学**

2011-01-20 @ NEC

Outline

標準形はお好き？

例）線形計画 (LP)

主問題

$$\begin{aligned} (\mathrm{P}) \quad \min \quad & \boldsymbol{c}^\top \boldsymbol{x}, \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq 0. \end{aligned}$$

双対問題

$$\begin{aligned} (\mathrm{D}) \quad \max \quad & \boldsymbol{b}^\top \boldsymbol{y}, \\ \text{s.t.} \quad & \boldsymbol{A}^\top \boldsymbol{y} \leq \boldsymbol{c}. \end{aligned}$$

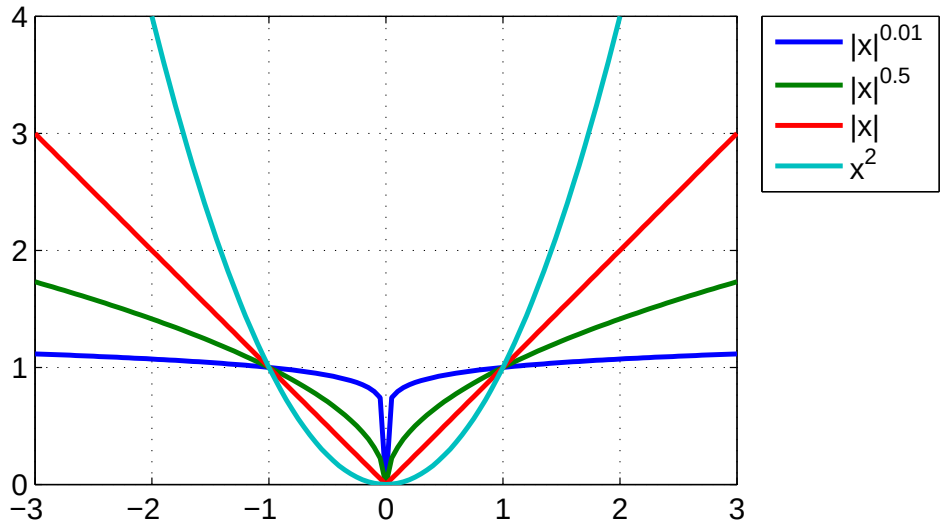
2 次計画 (QP)，**2 次錐計画** (SOCP)，**半正定値計画** (SDP)，etc...

- **良い点：既存の（ある程度）高性能なソルバーが利用できる．**
- **悪い点：問題のモデル化に制限，問題の書き換えが必要．**

モデル化に制限が生じる例：ℓ_p-正則化

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|^2 + \lambda \sum_{j=1}^{n} |w_j|^p.$$

- $p = 2$: **正則化最小二乗法**
  $\Rightarrow$ **逆行列で一発．**
- $1 < p < 2$: ?
- $p = 1$: Lasso $\Rightarrow$ **2 次計画．**

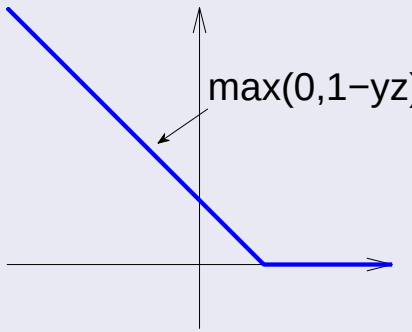


それでも標準形が好まれる理由（これを解決します）

1. **微分不可能性**: 微分不可能性 ＜ 制約付き最適化

SVM (微分不可能)

$$\min_{\boldsymbol{w}} \quad C\sum_{i=1}^{m} \ell_H(y_i \langle \boldsymbol{x}_i, \boldsymbol{w} \rangle) + \frac{1}{2}\|\boldsymbol{w}\|^2$$



SVM (2 次計画)

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & C\sum_{i=1}^{m} \xi_i + \frac{1}{2}\|\boldsymbol{w}\|^2, \\ \text{s.t.} \quad & y_i \langle \boldsymbol{x}_i, \boldsymbol{w} \rangle \geq 1 - \xi_i \\ & (i = 1, \ldots, m). \end{aligned}$$

（制約の数＝サンプル数）

2. **実装がめんどくさい**

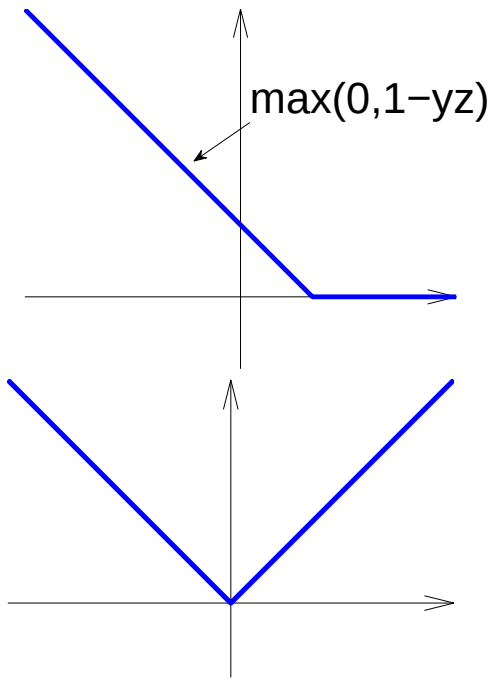
微分不可能性を持つ凸最適化問題の例

- SVM （**ロスが微分不可能**）

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad C\sum_{i=1}^{m}\ell_H(y_i\langle \boldsymbol{x}_i, \boldsymbol{w}\rangle) + \frac{1}{2}\|\boldsymbol{w}\|^2$$

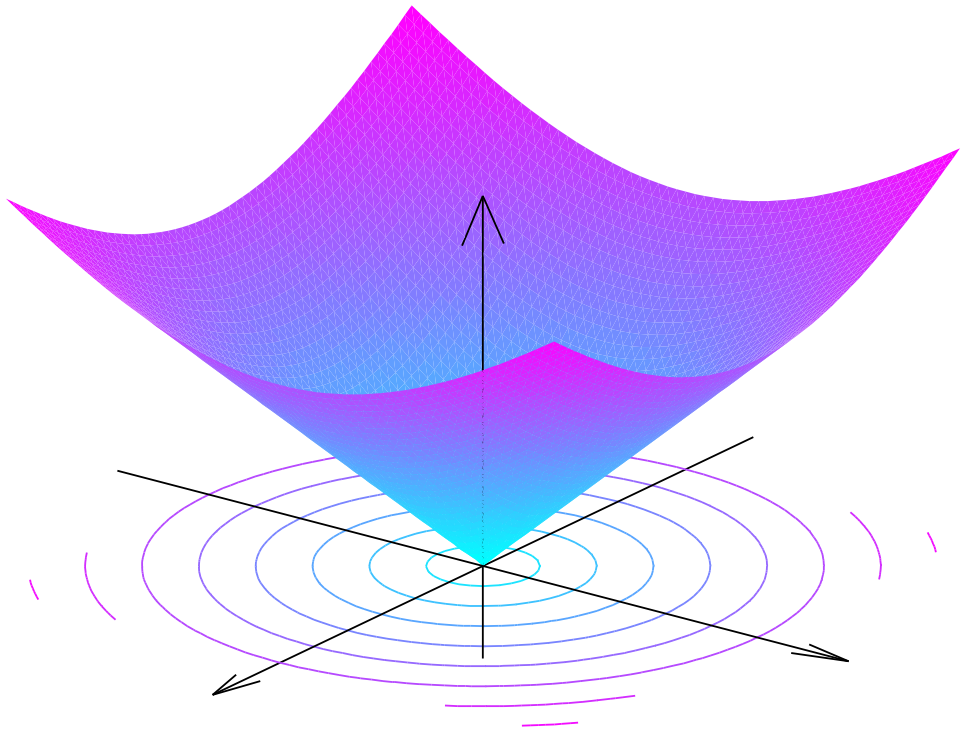
- Lasso（**正則化項が微分不可能**）

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad L(\boldsymbol{w}) + \lambda\sum_{j=1}^{n}|w_j|$$



マルチタスク学習 (Evgeniou et al 05)

$$\underset{\boldsymbol{w}_1,\boldsymbol{w}_2,\boldsymbol{w}_{12}}{\text{minimize}} \quad \underbrace{L(\boldsymbol{w}_1+\boldsymbol{w}_{12})}_{\text{タスク 1 のロス}}+\underbrace{L(\boldsymbol{w}_2+\boldsymbol{w}_{12})}_{\text{タスク 2 のロス}}+\lambda(\|\boldsymbol{w}_1\|+\|\boldsymbol{w}_2\|+\|\boldsymbol{w}_{12}\|)$$



マルチカーネル学習 (Bach et al 05; Suzuki & Tomioka 09)

- M 個の情報源があり，それぞれについて入力サンプル x_i と x_j の内積が $k_m(x_i, x_j)$ $(m = 1, \ldots, M)$ と計算できるとき，それらをいかに組み合わせて予測するか？
- 最適化問題

$$\underset{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_M}{\text{minimize}} \quad C \sum_{i=1}^{N} \ell_H \left(y_i \sum_{m=1}^{M} f_m(x_i) \right) + \lambda \sum_{m=1}^{M} \|\boldsymbol{f}_m\|$$

- それぞれの情報源について，表現定理より予測関数

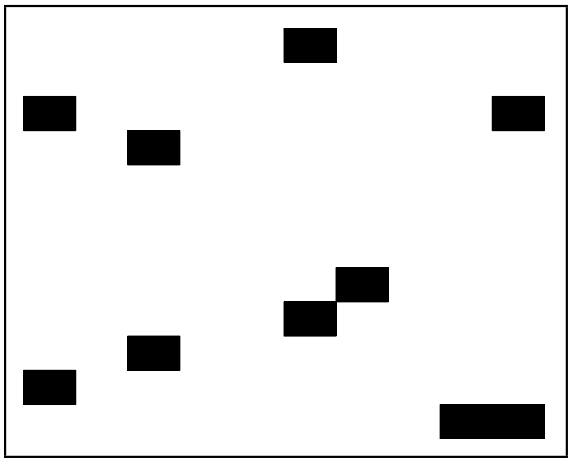
$$f_m(x_i) = \sum_{j=1}^{N} k_m(x_i, x_j) \alpha_j^{(m)}.$$

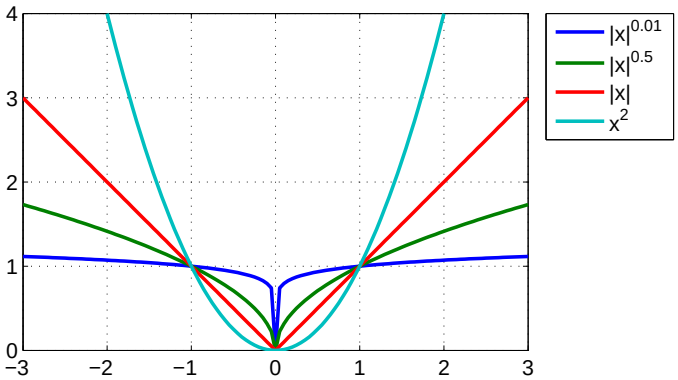
- 正則化項だけでなくロスも微分不可能．

行列穴埋め

- 部分的に観測された行列の未観測部分を低ランク性の仮定を使って予測する問題． $\Rightarrow$ 協調フィルタリング．

$$\underset{\boldsymbol{X}}{\text{minimize}} \quad \frac{1}{2}\|\Omega(\boldsymbol{X}-\boldsymbol{Y})\|^2 + \lambda \underbrace{\sum_{j=1}^{r}\sigma_j(\boldsymbol{X})}_{\text{特異値の線形和}} .$$





- Ω は見えている部分だけを取り出す操作．

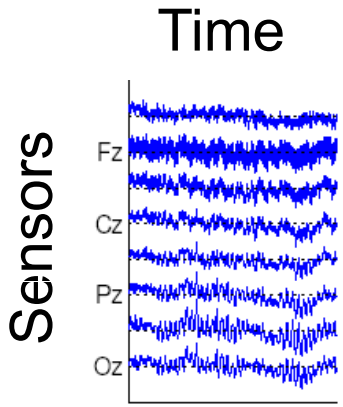
行列の空間上の判別問題

判別ラベル **予測関数**

$$y \in \{-1, +1\} \quad \Leftarrow \quad f(X) = \langle W, X \rangle + b$$

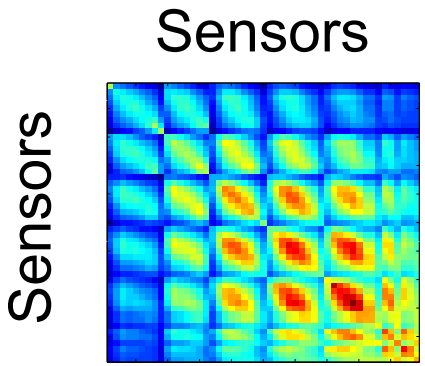
- Multivariate Time Series

$$X =$$

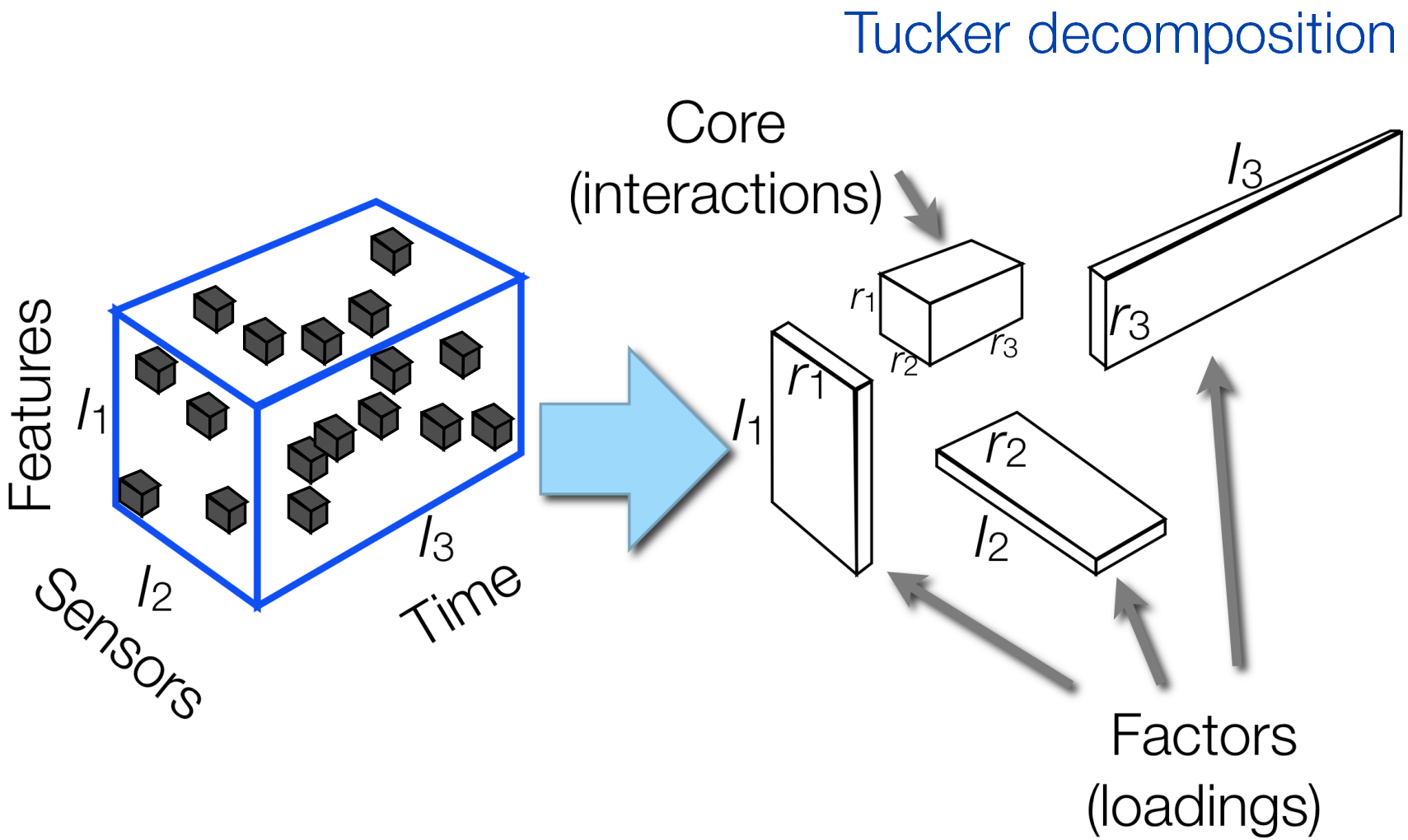


- Second order statistics

$$X =$$



テンソルの穴埋め


Tucker decomposition
Core
(interactions)
Features
Sensors
Time
I_1
I_2
I_3
r_1
r_2
r_3
Factors
(loadings)

アジェンダ

（割と）簡単に微分不可能な最適化問題を解くテクニック

- Prox **作用素** (proximity operator)
- **凸関数の共役** (Legendre **変換**)
- Operator splitting

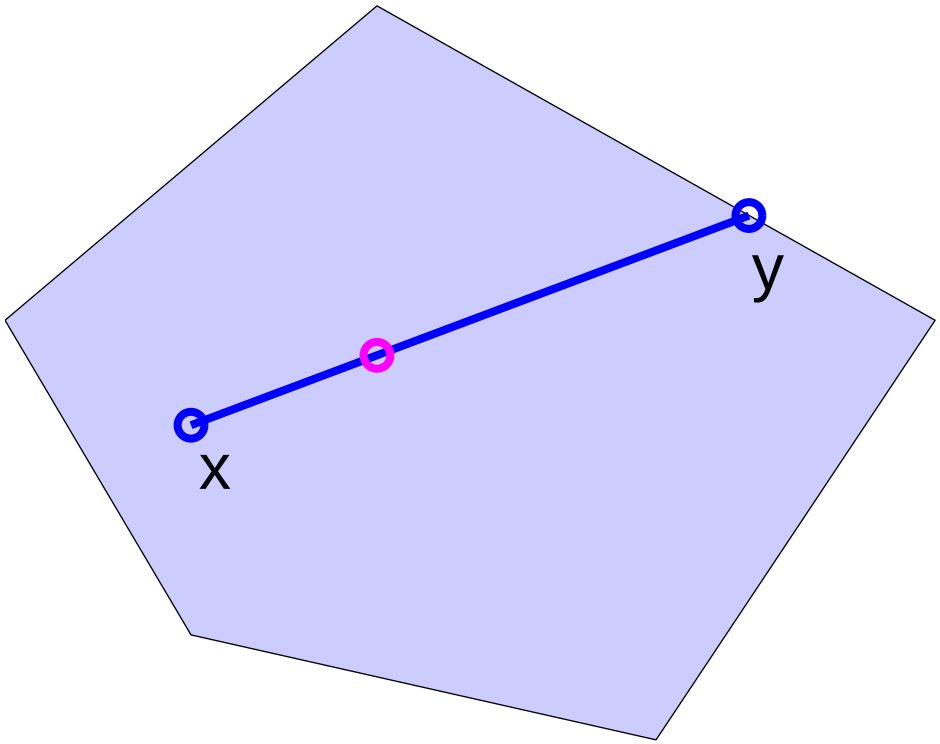
Outline

凸集合

$\mathbb{R}^n$ の部分集合 V は凸集合

$\Leftrightarrow$ V の任意の 2 点 $\boldsymbol{x}, \boldsymbol{y}$ を結ぶ線分が V に入っている．つまり

$$\forall \boldsymbol{x}, \boldsymbol{y} \in V, \forall \lambda \in [0,1], \quad \lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y} \in V.$$



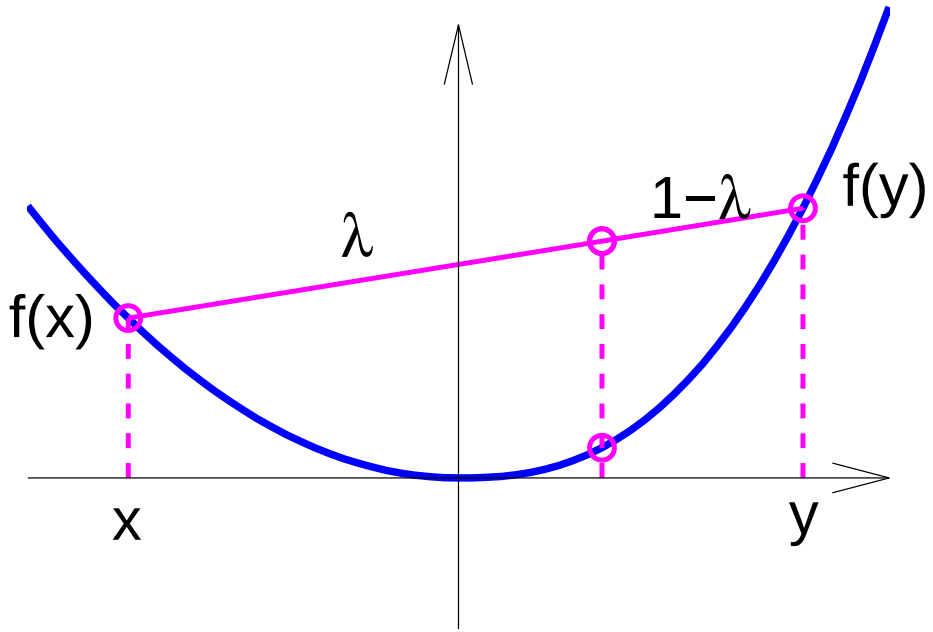
凸関数

関数 $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ **が（拡張値）凸関数**

$\Leftrightarrow$ **関数** f **の“ 弦 ”がつねに関数自身より上にある．つまり**

$$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \forall \lambda \in [0,1], \quad f((1-\lambda)\boldsymbol{x} + \lambda \boldsymbol{y}) \leq (1-\lambda) f(\boldsymbol{x}) + \lambda f(\boldsymbol{y})$$

$<$ **が成り立つ場合を** strictly convex **と言う．**



拡張値凸関数 - なぜ $+\infty$ を許すか

- $f(\boldsymbol{x}) = +\infty$ を許すと定義域（or 制約）を含めて扱える．
  例）$1/x$ は $x > 0$ で凸，

$$f(x) = \begin{cases} 1/x & (x > 0), \\ +\infty & (\text{otherwise}). \end{cases}$$

  と定義すればよい．

定義（定義関数）

凸集合 C の定義関数 δ_C を以下のように定義する．

$$\delta_C(\boldsymbol{x}) = \begin{cases} 0 & (\boldsymbol{x} \in C), \\ +\infty & (\text{otherwise}). \end{cases}$$

このように定義した δ_C は凸関数．

凸最適化問題

- **拡張値凸関数 f として**

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}).$$

- **制約付き問題**

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}) + \delta_C(\boldsymbol{x}).$$

- **正則化付き最小化問題**

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad \underbrace{L(\boldsymbol{x}) + \phi_\lambda(\boldsymbol{x})}_{=:f(\boldsymbol{x})}.$$

Outline

Proximal Minimization (Rockafellar 76)

1. $\boldsymbol{x}^0$ を適当に初期化する．
2. 以下を繰り返す．

$$\boldsymbol{x}^{t+1} = \operatorname*{argmin}_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \frac{1}{2\eta_t} \|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right)$$

- **良い点**: f が凸関数であれば必ず収束．しかも速い（超 1 次）

$$\|\boldsymbol{x}^{t+1} - \boldsymbol{x}^*\| \leq \frac{1}{1 + \sigma\eta_t} \|\boldsymbol{x}^t - \boldsymbol{x}^*\|$$

(Tomioka et al. 11)

- **悪い点**: そもそも $f(\boldsymbol{x})$ の最小化が難しい時，上の最小化を各ステップやるのはつらい．

勾配法

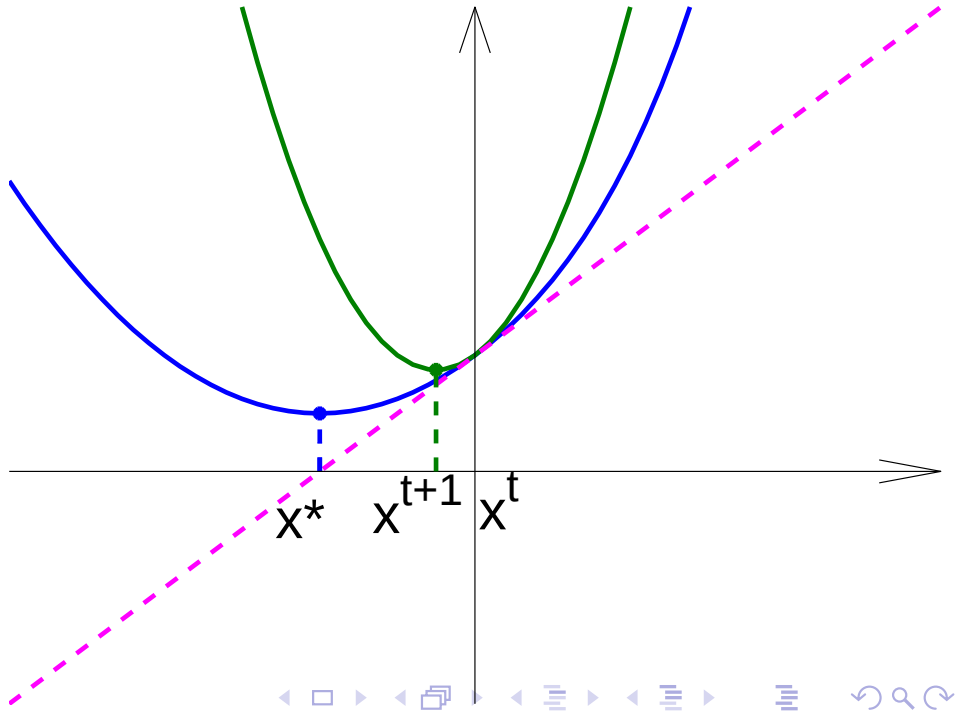
各ステップで $f(\boldsymbol{x})$ を $\boldsymbol{x}^t$ の周りで線形化して Proximal Minimization **する．**

$$\begin{aligned}\boldsymbol{x}^{t+1} &= \operatorname*{argmin}_{\boldsymbol{x}} \left(\nabla f(\boldsymbol{x}^t)(\boldsymbol{x} - \boldsymbol{x}^t) + \frac{1}{2\eta_t} \|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right) \\ &= \boldsymbol{x}^t - \eta_t \nabla f(\boldsymbol{x}^t)\end{aligned}$$

- **安定性の条件**: $\eta_t \leq 1/L(f)$.
- $L(f)$ **は微分** ∇f **のリプシッツ定数**:

  $$\|\nabla f(\boldsymbol{y}) - \nabla f(\boldsymbol{x})\| \leq L(f)\|\boldsymbol{y} - \boldsymbol{x}\|.$$

- f **が** 2 **階微分可能な場合は** $L(f)$**= ヘシアンの最大固有値の上限．**



0G: 射影付き勾配法 (Bertsekas 99; Nesterov 03)

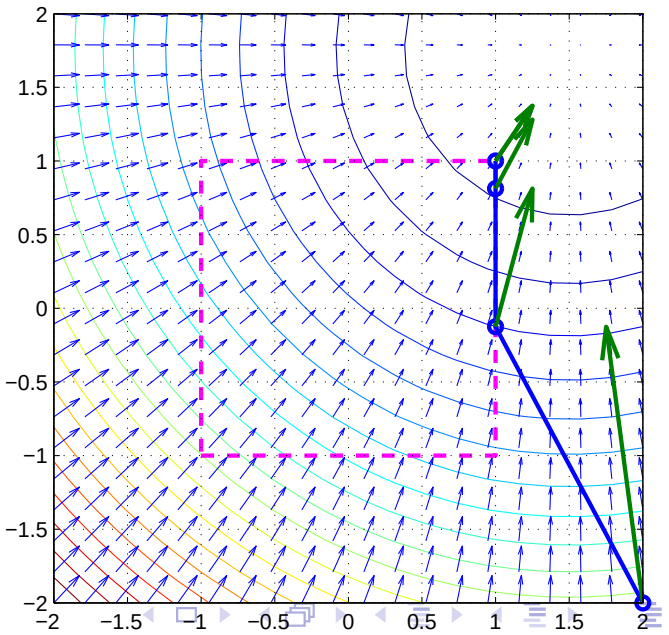
目的関数を線形化，δ_C は束縛条件の定義関数，

$$\begin{aligned}\boldsymbol{x}^{t+1} &= \operatorname*{argmin}_{\boldsymbol{x}} \left(\nabla f(\boldsymbol{x}^t)(\boldsymbol{x} - \boldsymbol{x}^t) + \delta_C(\boldsymbol{x}) + \frac{1}{2\eta_t}\|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right) \\ &= \operatorname*{argmin}_{\boldsymbol{x}} \left(\delta_C(\boldsymbol{x}) + \frac{1}{2\eta_t}\|\boldsymbol{x} - (\boldsymbol{x}^t - \eta_t \nabla f(\boldsymbol{x}^t))\|^2 \right) \\ &= \mathrm{proj}_C(\boldsymbol{x}^t - \eta_t \nabla f(\boldsymbol{x}^t)).\end{aligned}$$

- **安定性の条件は制約のない場合と同様**: $\eta_t \leq 1/L(f)$.
- **収束の速さ**

$$f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*) \leq \frac{L(f)\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|^2}{2k}$$

- **もちろん射影 proj_C が効率的に計算できることが必要．**



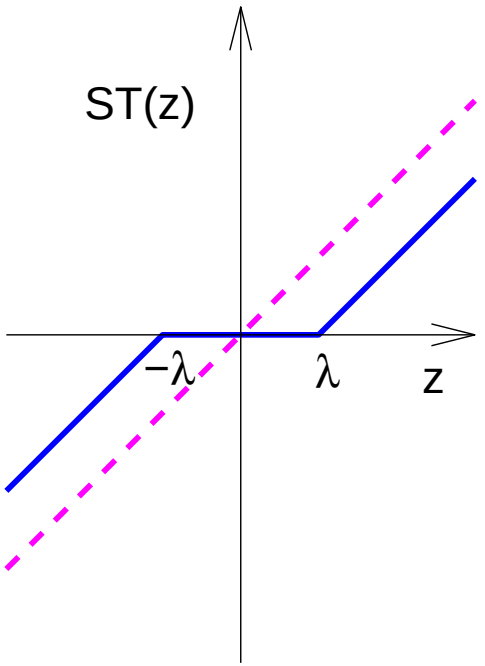
Proximal Operator: 射影の一般化

$$\mathrm{prox}_{\phi_\lambda}(\boldsymbol{z}) = \operatorname*{argmin}_{\boldsymbol{x}} \left(\phi_\lambda(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{z}\|^2 \right)$$

- **凸集合への射影**: $\mathrm{proj}_C(\boldsymbol{z}) = \mathrm{prox}_{\delta_C}(\boldsymbol{z})$.
- Soft-Threshold ($\phi_\lambda(\boldsymbol{x}) = \lambda\|\boldsymbol{x}\|_1$)

$$\mathrm{prox}_\lambda(\boldsymbol{z}) = \operatorname*{argmin}_{\boldsymbol{x}} \left(\lambda\|\boldsymbol{x}\|_1 + \frac{1}{2}\|\boldsymbol{x}-\boldsymbol{z}\|^2 \right)$$

$$= \begin{cases} z_j + \lambda & (z_j < -\lambda), \\ 0 & (-\lambda \le z_j \le \lambda), \\ z_j - \lambda & (z_j > \lambda). \end{cases}$$



- **何らかの意味で分離可能な ϕ_λ は Prox が簡単に計算できる.**
- **微分不可能でも解析的に計算できる.**

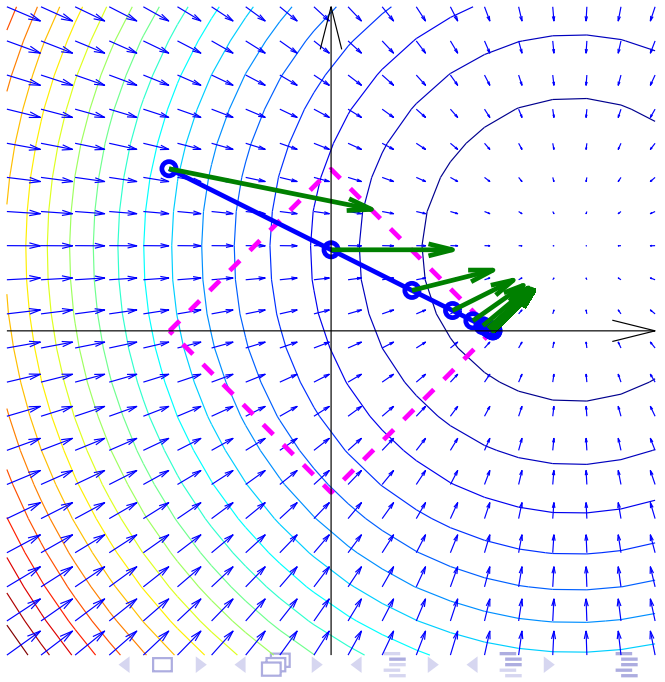
1G: Iterative Shrinkage Thresholding (IST)

$$\begin{aligned}\boldsymbol{x}^{t+1} &= \operatorname*{argmin}_{\boldsymbol{x}} \left(\nabla L(\boldsymbol{x}^t)(\boldsymbol{x} - \boldsymbol{x}^t) + \phi_\lambda(\boldsymbol{x}) + \frac{1}{2\eta_t}\|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right) \\ &= \operatorname*{argmin}_{\boldsymbol{x}} \left(\phi_\lambda(\boldsymbol{x}) + \frac{1}{2\eta_t}\|\boldsymbol{x} - (\boldsymbol{x}^t - \eta_t \nabla L(\boldsymbol{x}^t))\|^2 \right) \\ &= \mathrm{prox}_{\lambda\eta_t}(\boldsymbol{x}^t - \eta_t \nabla L(\boldsymbol{x}^t)).\end{aligned}$$

- **安定性の条件，収束性は射影付き勾配法と同じ．** (Beck & Teboulle 09)

$$f(\boldsymbol{x}^k) - f(\boldsymbol{x}^*) \leq \frac{L(f)\|\boldsymbol{x}_0 - \boldsymbol{x}^*\|^2}{2k}$$

- Prox **作用素** prox_λ **が計算できれば実装簡単．**
- Forward-Backward Splitting **とも呼ばれる** (Lions & Mercier 76)



IST を用いた行列穴埋め (Mazumder et al. 10)

ロス:

$$L(\boldsymbol{X}) = \frac{1}{2}\|\Omega(\boldsymbol{X} - \boldsymbol{Y})\|^2.$$

微分:

$$\nabla L(\boldsymbol{X}) = \Omega^\top(\Omega(\boldsymbol{X} - \boldsymbol{Y}))$$

正則化項:

$$\phi_\lambda(\boldsymbol{X}) = \lambda \sum_{j=1}^{r} \sigma_j(\boldsymbol{X}) \quad (\textbf{特異値の線形和}).$$

Prox **作用素** (Singular Value Thresholding):

$$\mathrm{prox}_\lambda(\boldsymbol{Z}) = \boldsymbol{U}\max(\boldsymbol{S} - \lambda \boldsymbol{I}, 0)\boldsymbol{V}^\top.$$

反復式:

$$\boldsymbol{X}^{t+1} = \mathrm{prox}_{\lambda\eta_t}\left((\boldsymbol{I} - \eta_t\Omega^\top\Omega)(\boldsymbol{X}^t) + \eta_t\Omega^\top\Omega(\boldsymbol{Y}^t)\right)$$

- $\eta_t = 1$ **の時，予測値で穴埋め，観測値で上書き，**Soft-Threshold **かける，の繰り返し．**

FISTA: IST の加速版 (Beck & Teboulle 09; Nesterov 07)

1. $\boldsymbol{x}^0$ を適当に初期化，$\boldsymbol{y}^1 = \boldsymbol{x}^0$，$s_1 = 1$ とする．
2. $\boldsymbol{x}^t$ を更新:

$$\boldsymbol{x}^t = \mathrm{prox}_{\lambda\eta_t}(\boldsymbol{y}^t - \eta_t \nabla L(\boldsymbol{y}^t)).$$

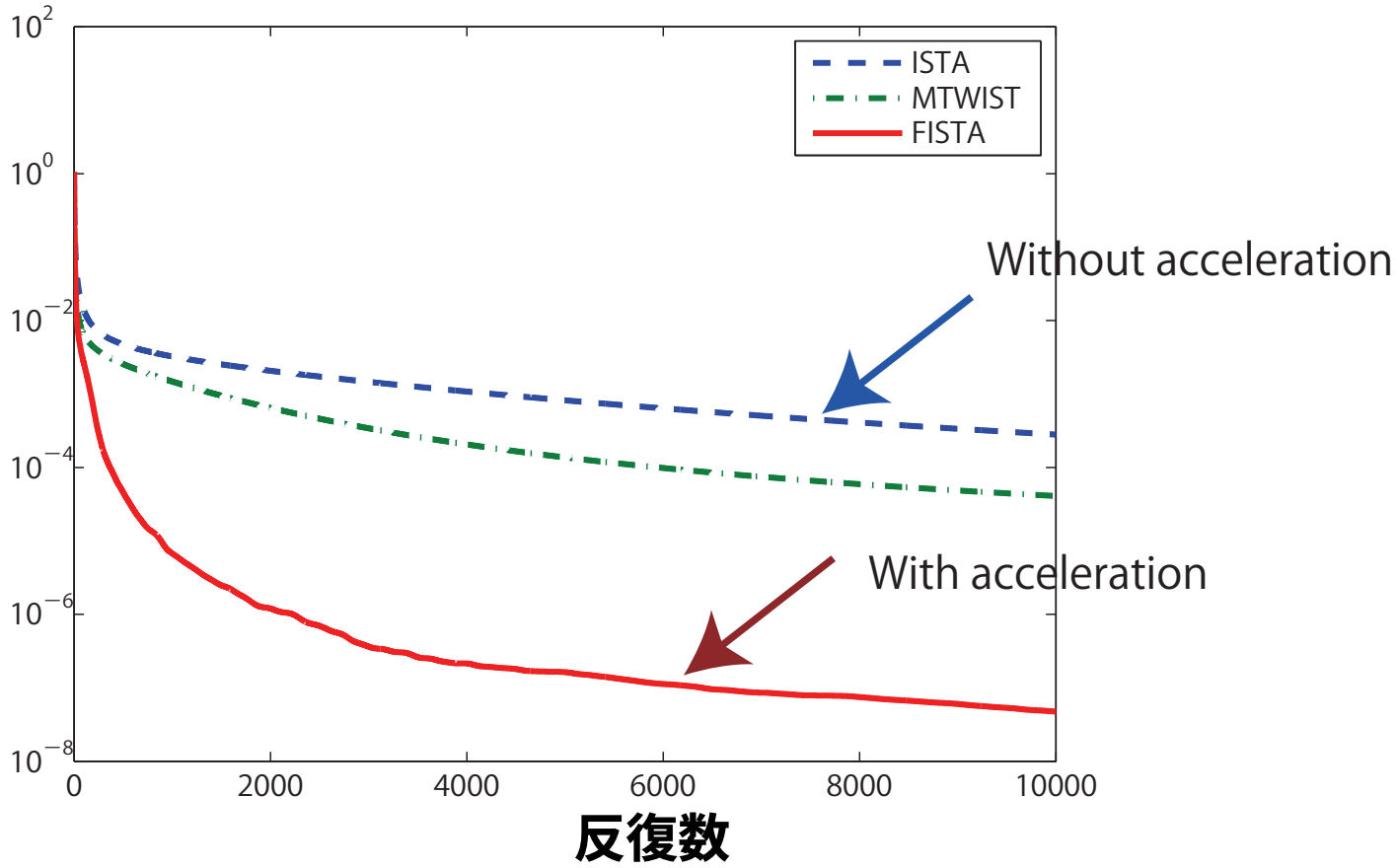
3. $\boldsymbol{y}^t$ を更新:

$$\boldsymbol{y}^{t+1} = \boldsymbol{x}^t + \left(\frac{s_t - 1}{s_{t+1}}\right)(\boldsymbol{x}^t - \boldsymbol{x}^{t-1}),$$

ただし，$s_{t+1} = \left(1 + \sqrt{1 + 4s_t^2}\right)/2$.

- ステップあたりの計算量は IST とほぼ同じ．収束性は $O(1/k^2)$.
- どこの点で勾配ステップを取るべきか予測している感じ？

加速の効果

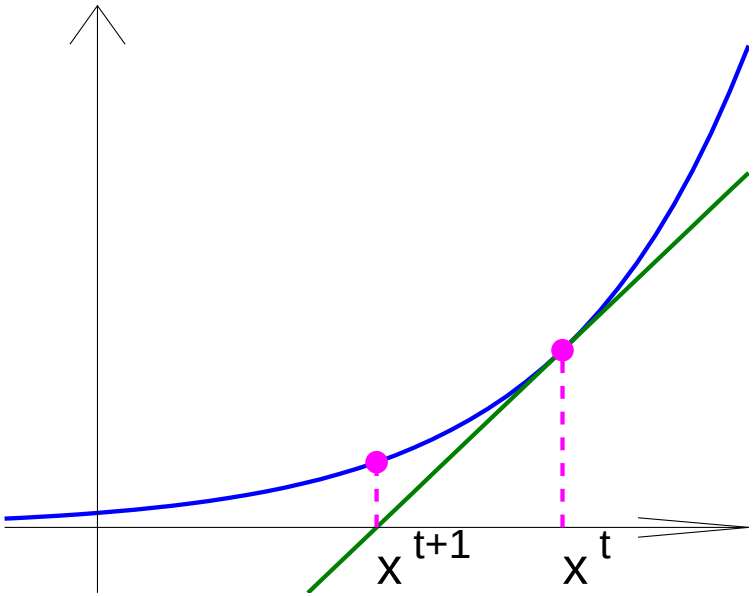


Beck & Teboulle 2009 SIAM J. IMAGING SCIENCES Vol. 2, No. 1, pp. 183-202 より

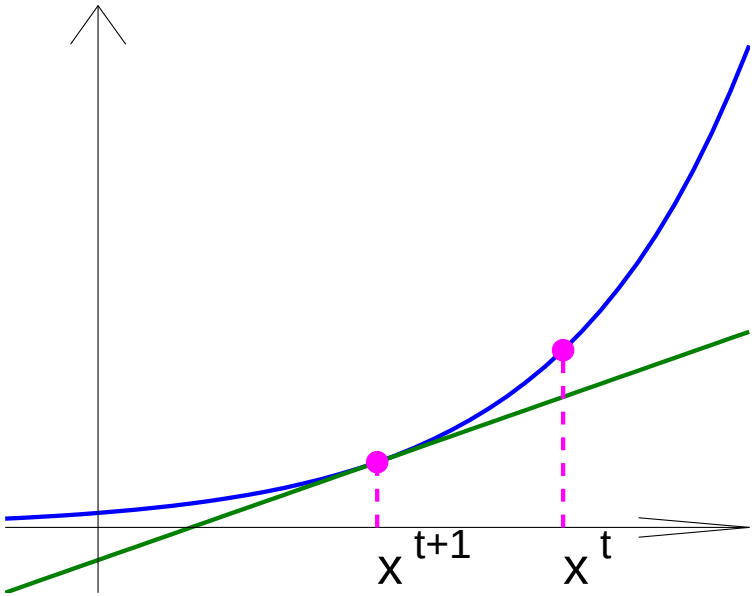
3G?: もっとタイトな下限を作るアイディア

IST:

$$\boldsymbol{x}^{t+1} = \operatorname*{argmin}_{\boldsymbol{x}} \left(\nabla L(\boldsymbol{x}^t)(\boldsymbol{x} - \boldsymbol{x}^t) + \phi_\lambda(\boldsymbol{x}) + \frac{1}{2\eta_t} \|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right)$$



IST: **現在の点 $\boldsymbol{x}^t$ で下限を作る．**



DAL: **次の点 $\boldsymbol{x}^{t+1}$ で下限を作る．**

次の点で下限を作るには？

下限の傾き y をパラメータとして最適化する:

$$\begin{aligned}\boldsymbol{x}^{t+1} &= \operatorname*{argmin}_{\boldsymbol{x}} \left(L(\boldsymbol{x}) + \phi_\lambda(\boldsymbol{x}) + \frac{1}{2\eta_t}\|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right) \\ &= \operatorname*{argmin}_{\boldsymbol{x}} \left(\max_{\boldsymbol{y}}(\langle \boldsymbol{x}, \boldsymbol{y}\rangle - L^*(\boldsymbol{y})) + \phi_\lambda(\boldsymbol{x}) + \frac{1}{2\eta_t}\|\boldsymbol{x} - \boldsymbol{x}^t\|^2 \right)\end{aligned}$$

$\Downarrow$ **Min と Max の順番を入れ替えて計算する**

DAL (Tomioka et al 11)

$$\boldsymbol{x}^{t+1} = \operatorname{prox}_{\lambda\eta_t}\left(\boldsymbol{x}^t - \eta_t \boldsymbol{y}^t\right),$$

ただし，

$$\boldsymbol{y}^t = \operatorname*{argmax}_{\boldsymbol{y}} \left(-L^*(\boldsymbol{y}) - \frac{1}{\eta_t}\Phi^*_{\lambda\eta_t}\left(\boldsymbol{x}^t - \eta_t \boldsymbol{y}\right) \right)$$

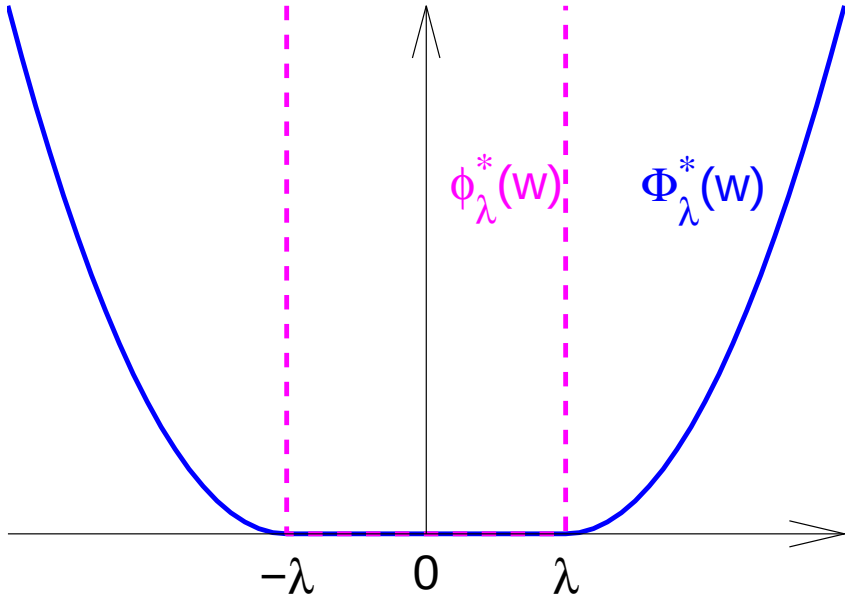
DAL の利点 (ℓ_1-正則化の場合)

(1) Prox 作用素は解析的に計算可能

$$\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t \lambda}\left(\boldsymbol{x}^t - \eta_t \boldsymbol{y}^t\right)$$

(2) 内部最適化は微分可能

$$\boldsymbol{y}^t = \underset{\boldsymbol{y}}{\mathrm{argmax}}\left(\underbrace{-L^*(\boldsymbol{y})}_{\text{微分可能}} - \frac{1}{2\eta_t}\underbrace{\|\mathrm{prox}_{\lambda\eta_t}(\boldsymbol{x}^t - \eta_t \boldsymbol{y})\|^2}_{\text{非ゼロ成分の数に比例}}\right)$$



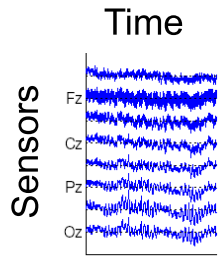
DAL を用いた行列の空間上の判別問題

判別ラベル　　**予測関数**

$$\boldsymbol{y} \in \{-1, +1\} \quad \Leftarrow \quad f(\boldsymbol{X}) = \langle \boldsymbol{W}, \boldsymbol{X} \rangle + b$$

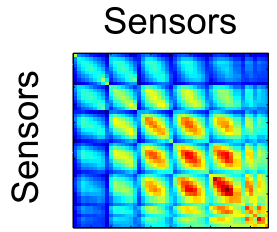
- Multivariate Time Series

$$X =$$



- Second order statistics

$$X =$$



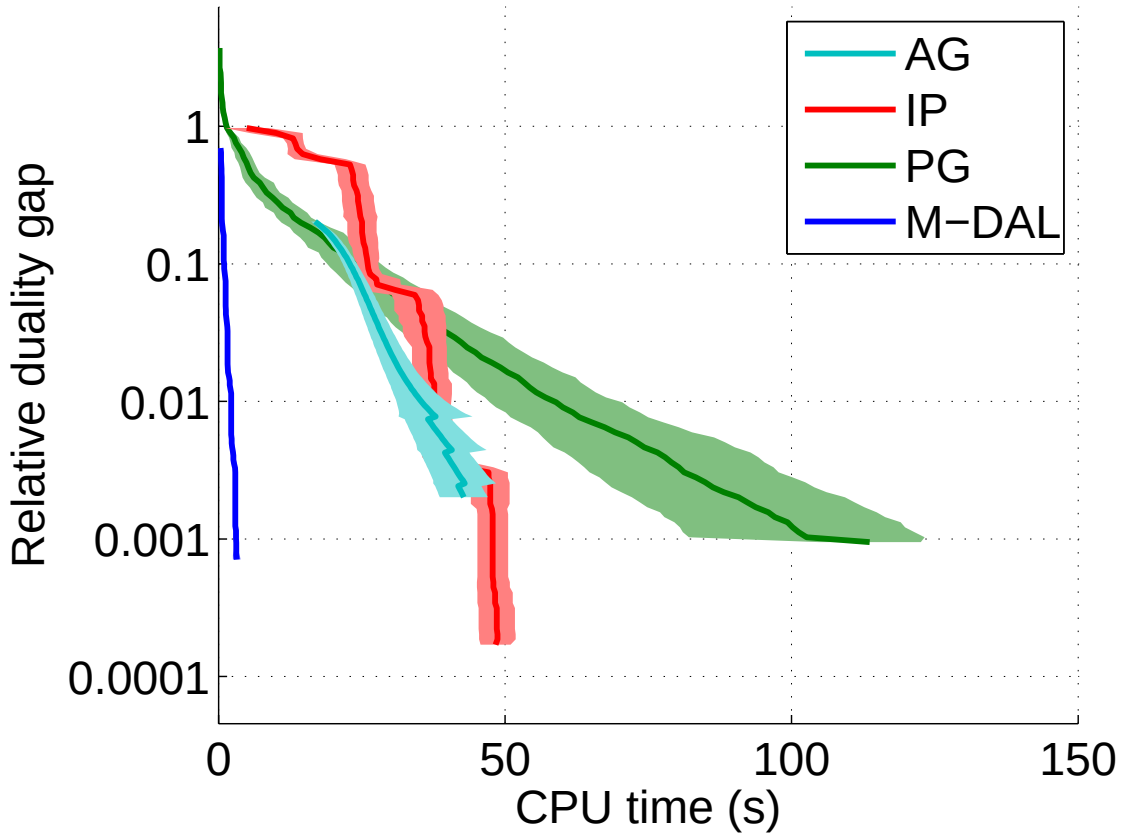
最適化問題:

$$\underset{\boldsymbol{W}}{\text{minimize}} \quad \sum_{i=1}^{m} \ell_{\text{LR}}(y_i f(\boldsymbol{X}_i)) + \lambda \underbrace{\|\boldsymbol{W}\|_{\text{tr}}}_{\text{低ランク化}}$$

ロジスティック損失:

$$\ell_{\text{LR}}(z_i) = \log(1 + \exp(-z_i))$$

DAL を用いた行列の空間上の判別問題 (Tomioka et al 10)



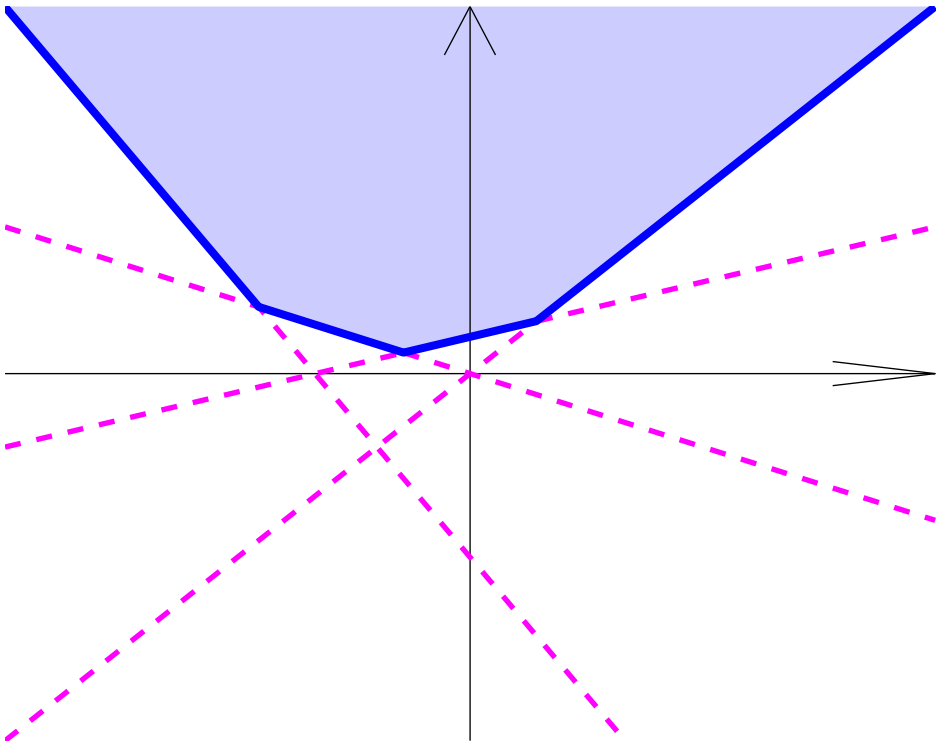
- AG: **加速付き** IST
- IP: **内点法**
- PG: **射影勾配法**
- M-DAL: **提案法**

Outline

凸関数の作り方

線形関数の point-wise maximum **は凸関数.**

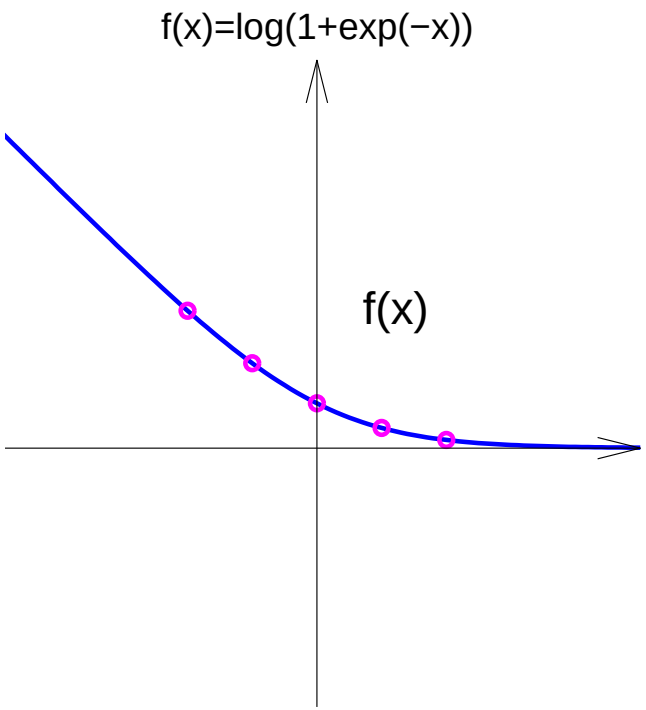
$$f(\boldsymbol{x}) = \max_{i=1,\dots,k} \left(\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle - b_i \right).$$

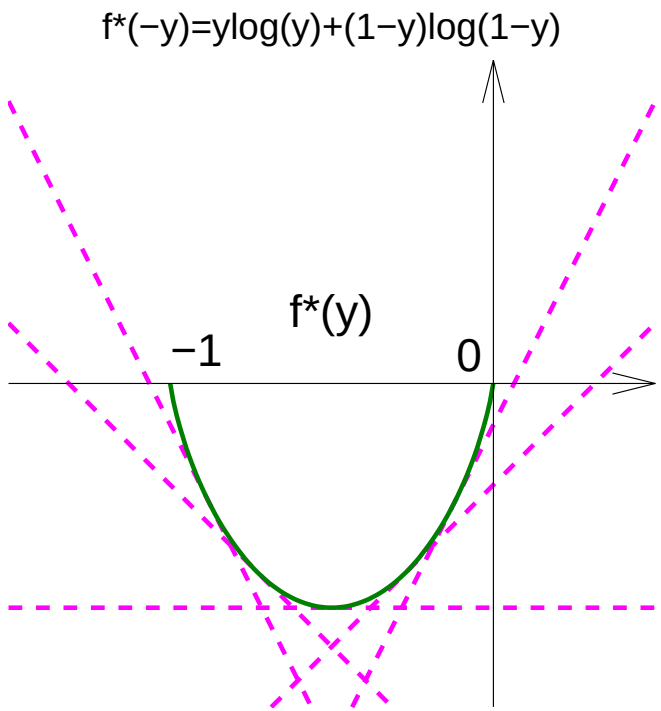


凸共役 (Fenchel-Legendre 変換)

$$f^*(\boldsymbol{y}) = \sup_{\boldsymbol{x}\in\mathbb{R}^n} \left(\langle \boldsymbol{y}, \boldsymbol{x}\rangle - f(\boldsymbol{x})\right).$$

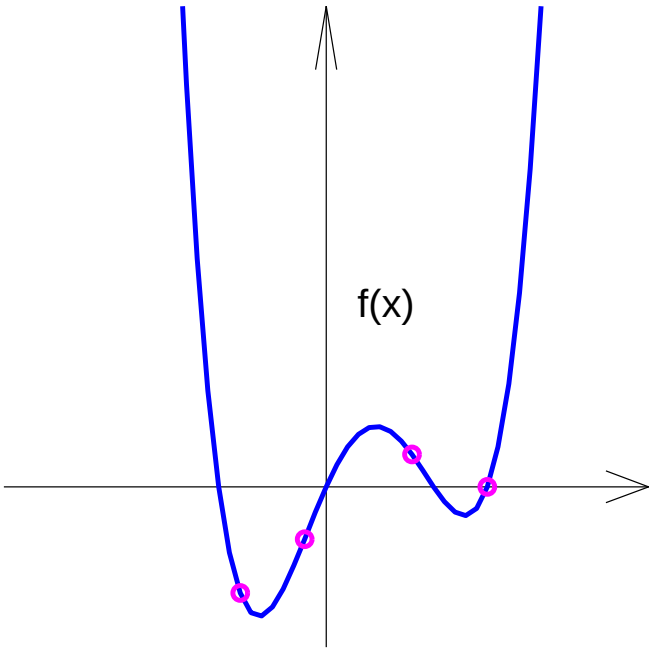
のように定義される f^* を関数 f の凸共役と呼ぶ.

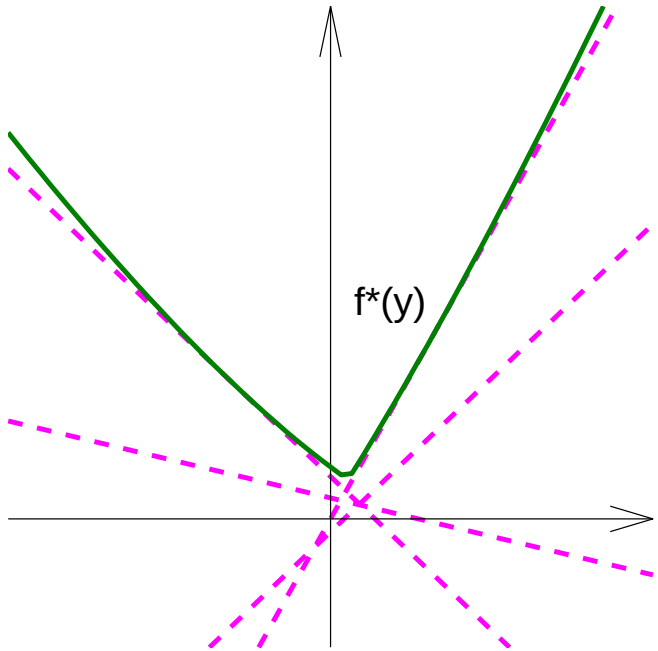




注意: $f(x)$ は凸でなくてもよい

$f(x)$ が凸関数であってもなくても $f^*(y)$ は凸関数.





凸共役の性質

定義より

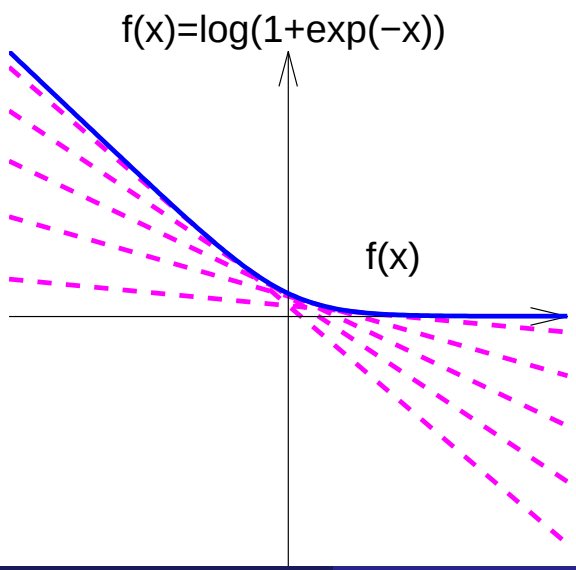
$$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \quad f(\boldsymbol{x}) + f^*(\boldsymbol{y}) \geq \langle \boldsymbol{y}, \boldsymbol{x} \rangle .$$

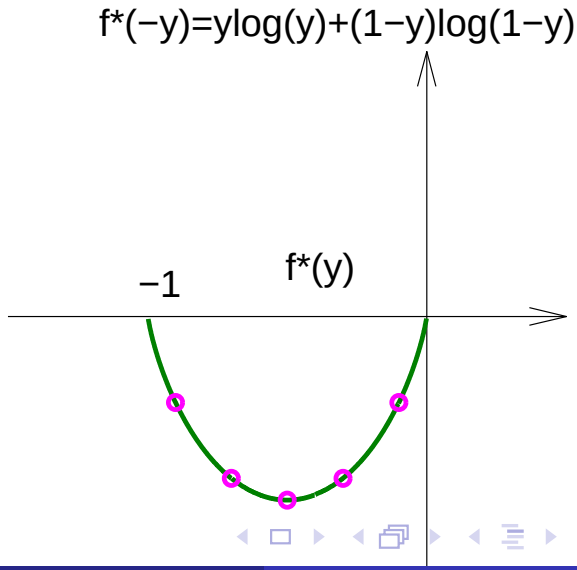
$\boldsymbol{x}$ がある $\boldsymbol{y}$ に関して $\langle \boldsymbol{y}, \boldsymbol{x} \rangle - f(\boldsymbol{x})$ を最大化するならば

$$f^*(\boldsymbol{y}) = \langle \boldsymbol{y}, \boldsymbol{x} \rangle - f(\boldsymbol{x}).$$

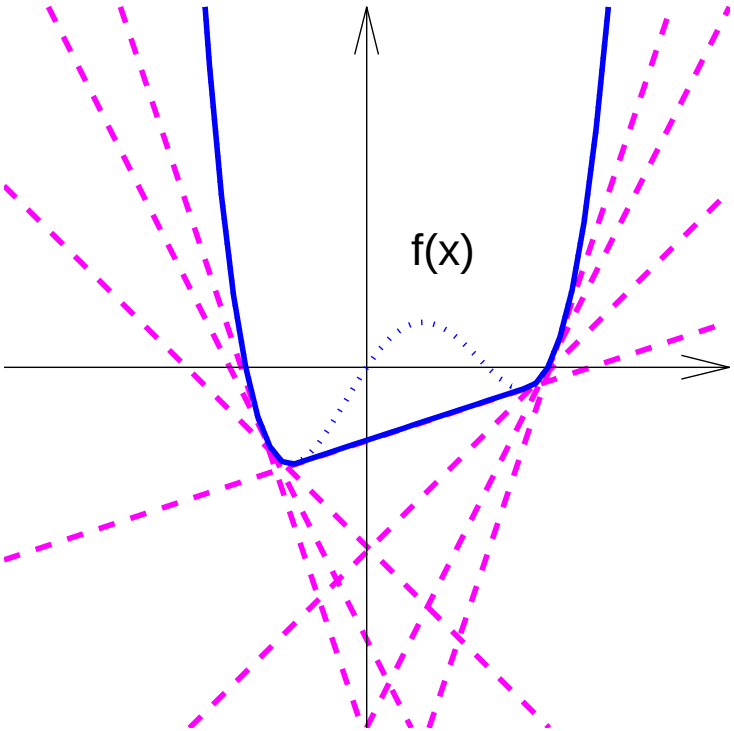
つまり，

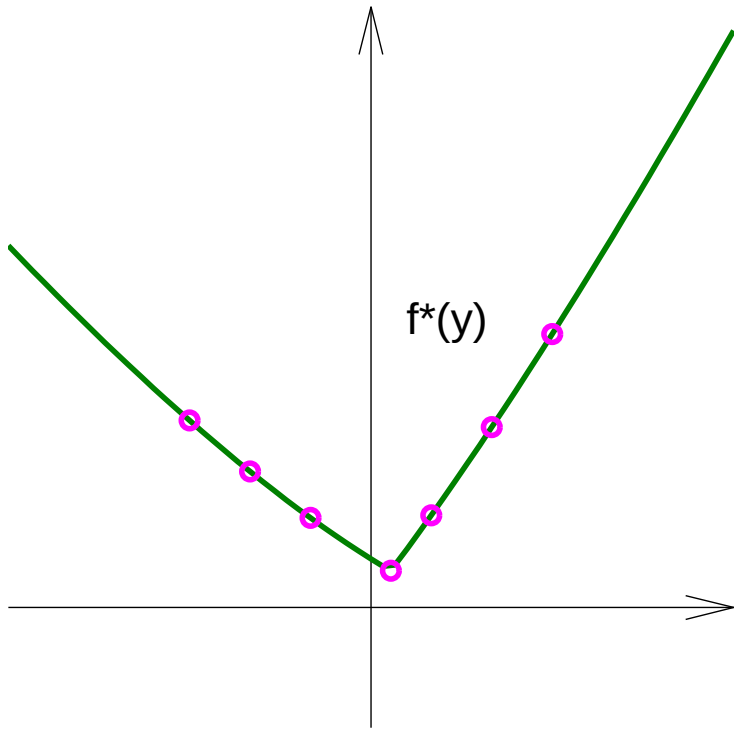
$$f(\boldsymbol{x}) = f^{**}(\boldsymbol{x}) = \sup_{\boldsymbol{y}} \left(\langle \boldsymbol{y}, \boldsymbol{x} \rangle - f^*(\boldsymbol{y}) \right).$$





凸でない f の場合 $f \neq f^{**}$





- f^* **が微分可能** $\Leftrightarrow$ f **が** strictly convex.

Uzawa's method (Dual ascent; 双対分解)

最適化問題（あえて制約付き問題として書く）:

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n,\boldsymbol{z}\in\mathbb{R}^m}{\text{minimize}} \quad & f(\boldsymbol{z}) + g(\boldsymbol{x}), \\ \text{subject to} \quad & \boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}. \end{aligned}$$

ラグランジアン:

$$L(\boldsymbol{x}, \boldsymbol{z}, \alpha) = f(\boldsymbol{z}) + g(\boldsymbol{x}) + \alpha^\top(\boldsymbol{z} - \boldsymbol{A}\boldsymbol{x}).$$

ラグランジアンの最小化は凸共役の計算

$$\begin{aligned} -f^*(-\alpha) &= \min_{\boldsymbol{z}} \left(f(\boldsymbol{z}) + \left\langle \alpha^t, \boldsymbol{z} \right\rangle \right), \\ -g^*(\boldsymbol{A}^\top\alpha) &= \min_{\boldsymbol{x}} \left(g(\boldsymbol{x}) - \left\langle \boldsymbol{A}^\top\alpha^t, \boldsymbol{x} \right\rangle \right). \end{aligned}$$

Uzawa's method (Uzawa 58; Bertsekas 99)

ラグランジアンを x, z に関して最小化:

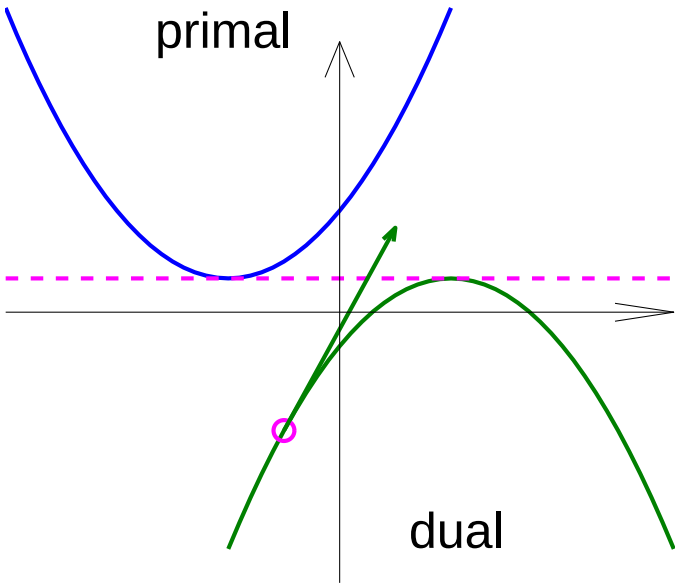
$$\boldsymbol{z}^{t+1} = \mathrm{argmin}_{\boldsymbol{z}} \left(f(\boldsymbol{z}) + \left\langle \alpha^t, \boldsymbol{z} \right\rangle \right),$$
$$\boldsymbol{x}^{t+1} = \mathrm{argmin}_{\boldsymbol{x}} \left(g(\boldsymbol{x}) - \left\langle \boldsymbol{A}^\top \alpha^t, \boldsymbol{x} \right\rangle \right).$$

ラグランジュ乗数 α^t を更新:

$$\alpha^{t+1} = \alpha^t + \eta_t (\boldsymbol{z}^{t+1} - \boldsymbol{A}\boldsymbol{x}^{t+1}).$$

宇澤 弘文

- **ラグランジュ乗数の更新は双対での勾配法** (dual ascent) **に対応**
- **良い点**: **シンプル！**
- **悪い点**: **凸共役 f^*, g^* が微分できないとき劣勾配法** (sub-gradient ascent) **収束するの？**



Uzawa's method を用いた行列穴埋め (Cai et al. 08)

$$\underset{\boldsymbol{X}}{\text{minimize}} \quad \underbrace{\frac{1}{2\lambda}\|\boldsymbol{z}-\boldsymbol{y}\|^2}_{\text{Strictly convex}} + \underbrace{\left(\tau\|\boldsymbol{X}\|_{\text{tr}} + \frac{1}{2}\|\boldsymbol{X}\|^2\right)}_{\text{Strictly convex}},$$

$$\text{subject to} \quad \Omega(\boldsymbol{X}) = \boldsymbol{z}.$$

$$\Downarrow$$

ラグランジアン:

$$L(\boldsymbol{X},\boldsymbol{z},\alpha) = \underbrace{\frac{1}{2\lambda}\|\boldsymbol{z}-\boldsymbol{y}\|^2}_{=f(\boldsymbol{z})} + \underbrace{\left(\tau\|\boldsymbol{X}\|_{\text{tr}} + \frac{1}{2}\|\boldsymbol{X}\|^2\right)}_{=g(\boldsymbol{x})} + \alpha^\top(\boldsymbol{z}-\Omega(\boldsymbol{X})).$$

Uzawa's method:

$$\begin{cases} \boldsymbol{X}^{t+1} = \text{prox}_\tau\left(\Omega^\top(\alpha^t)\right) & \text{(Singular-Value Thresholding)} \\ \boldsymbol{z}^{t+1} = \boldsymbol{y} - \lambda\alpha^t \\ \alpha^{t+1} = \alpha^t + \eta_t(\boldsymbol{z}^{t+1} - \Omega(\boldsymbol{X}^{t+1})) \end{cases}$$

Uzawa's method を用いた行列穴埋め (Cai et al. 08)

$$\underset{\boldsymbol{X}}{\text{minimize}} \quad \underbrace{\frac{1}{2\lambda}\|\boldsymbol{z}-\boldsymbol{y}\|^2}_{\text{Strictly convex}} + \underbrace{\left(\tau\|\boldsymbol{X}\|_{\text{tr}} + \frac{1}{2}\|\boldsymbol{X}\|^2\right)}_{\text{Strictly convex}},$$

$$\text{subject to} \quad \Omega(\boldsymbol{X}) = \boldsymbol{z}.$$

$$\Downarrow$$

ラグランジアン:

$$L(\boldsymbol{X},\boldsymbol{z},\boldsymbol{\alpha}) = \underbrace{\frac{1}{2\lambda}\|\boldsymbol{z}-\boldsymbol{y}\|^2}_{=f(\boldsymbol{z})} + \underbrace{\left(\tau\|\boldsymbol{X}\|_{\text{tr}} + \frac{1}{2}\|\boldsymbol{X}\|^2\right)}_{=g(\boldsymbol{x})} + \boldsymbol{\alpha}^\top(\boldsymbol{z}-\Omega(\boldsymbol{X})).$$

Uzawa's method:

$$\begin{cases} \boldsymbol{X}^{t+1} = \text{prox}_\tau\left(\Omega^\top(\boldsymbol{\alpha}^t)\right) & \text{(Singular-Value Thresholding)} \\ \boldsymbol{z}^{t+1} = \boldsymbol{y} - \lambda\boldsymbol{\alpha}^t \\ \boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + \eta_t(\boldsymbol{z}^{t+1} - \Omega(\boldsymbol{X}^{t+1})) \end{cases}$$

拡張ラグランジュ法 (Augmented Lagrangian Method)

拡張ラグランジアン

$$L_\eta(\boldsymbol{x}, \boldsymbol{z}, \alpha) = f(\boldsymbol{z}) + g(\boldsymbol{x}) + \alpha^\top(\boldsymbol{z} - \boldsymbol{A}\boldsymbol{x}) + \frac{\eta}{2}\|\boldsymbol{z} - \boldsymbol{A}\boldsymbol{x}\|^2.$$

拡張ラグランジュ法:

$$\begin{cases} \text{拡張ラグランジアンを } \boldsymbol{x}, \boldsymbol{z} \text{ に関して最小化:} \\ (\boldsymbol{x}^{t+1}, \boldsymbol{z}^{t+1}) = \underset{\boldsymbol{x}\in\mathbb{R}^n, \boldsymbol{z}\in\mathbb{R}^m}{\operatorname{argmin}} \; L_{\eta_t}(\boldsymbol{x}, \boldsymbol{z}, \alpha^t). \\ \\ \text{ラグランジュ乗数を更新:} \\ \alpha^{t+1} = \alpha^t + \eta_t(\boldsymbol{z}^{t+1} - \boldsymbol{A}\boldsymbol{x}^{t+1}). \end{cases}$$

- **良い点**: ペナルティ項を追加（双対は必ず微分可能になる）
- **悪い点**: $\boldsymbol{x}$ と $\boldsymbol{z}$ の間に絡みが発生!（別々に最小化できない）

実は

- **拡張ラグランジュ法をまじめにやると双対側で** Proximal Minimization

$$\boldsymbol{\alpha}^{t+1} = \underset{\boldsymbol{\alpha}\in\mathbb{R}^m}{\operatorname{argmin}} \Big(\underbrace{f^*(-\boldsymbol{\alpha}) + g^*(\boldsymbol{A}^\top \boldsymbol{\alpha})}_{\text{双対目的関数の符号反転}} + \frac{1}{2\eta_t} \|\boldsymbol{\alpha} - \boldsymbol{\alpha}^t\|^2 \Big)$$

  をやっているのと等価．
- DAL (Dual Augmented Lagrangian) **はその逆．**
- **拡張ラグランジュを不真面目に解くと双対側で色々な手法が出てくる．**

対応関係

	主問題	**双対**
厳密	**拡張ラグランジアン**	Proximal Minimization
	Proximal Minimization	DAL
近似	Alternating Minimization Algorithm (Tseng 91)	Forward-Backward Splitting (IST **と等価**)
	Alternating Direction Method of Multipliers (Gabay & Mercier 76)	Douglas-Rachford Splitting (Lions & Mercier 76)

Alternating Direction Method of Multipliers (ADMM; Gabay & Mercier 76)

$$
\begin{cases}
\textbf{拡張ラグランジアンを } x \textbf{ に関して最小化:} \\
\boldsymbol{x}^{t+1} = \underset{\boldsymbol{x} \in \mathbb{R}^n}{\operatorname{argmin}} L_{\eta_t}(\boldsymbol{x}, \boldsymbol{z}^t, \alpha^t). \\
\textbf{拡張ラグランジアンを } z \textbf{ に関して最小化:} \\
\boldsymbol{z}^{t+1} = \underset{\boldsymbol{z} \in \mathbb{R}^m}{\operatorname{argmin}} L_{\eta_t}(\boldsymbol{x}^{t+1}, \boldsymbol{z}, \alpha^t). \\
\textbf{ラグランジュ乗数を更新:} \\
\alpha^{t+1} = \alpha^t + \eta_t(\boldsymbol{z}^{t+1} - \boldsymbol{A}\boldsymbol{x}^{t+1}).
\end{cases}
$$

- **今更新した x^{t+1} が z^{t+1} の計算に入っているところがポイント.**

ADMM (Gabay & Mercier 76)

書き直すと

$$\begin{cases} \boldsymbol{x}^{t+1} = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \left(g(\boldsymbol{x}) + \dfrac{\eta_t}{2} \|\boldsymbol{z}^t - \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\alpha}^t/\eta_t\|^2 \right) \\ \boldsymbol{z}^{t+1} = \underset{\boldsymbol{z}\in\mathbb{R}^m}{\operatorname{argmin}} \left(f(\boldsymbol{z}) + \dfrac{\eta_t}{2} \|\boldsymbol{z} - \boldsymbol{A}\boldsymbol{x}^{t+1} + \boldsymbol{\alpha}^t/\eta_t\|^2 \right) \\ \boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + \eta_t(\boldsymbol{z}^{t+1} - \boldsymbol{A}\boldsymbol{x}^{t+1}). \end{cases}$$

- z に関する最小化は Prox 作用素 prox_f（簡単）.
- x に関する最小化は行列 A が変数を絡ませるのでちょっと難しい.
- 1 反復あたりのコストが同じなら FBS より明らかに速い（理論的には不明）
- 双対側での Douglas Rachford Splitting と等価 $\Rightarrow$ ステップサイズ η によらず ADMM は安定. (Lions & Mercier 76; Eckstein & Bertsekas 92)

ADMM (Gabay & Mercier 76)

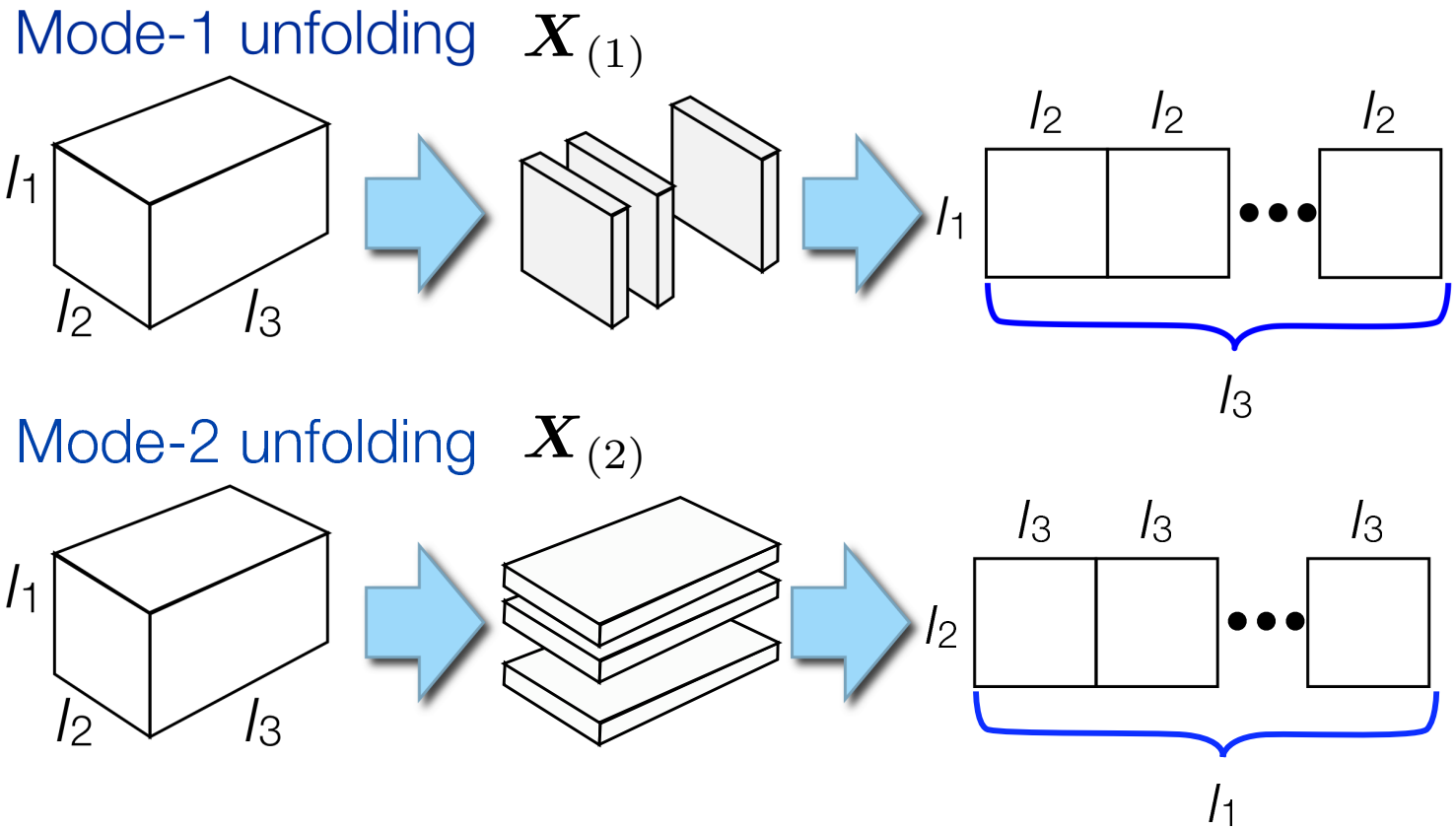
書き直すと

$$\begin{cases} \boldsymbol{x}^{t+1} = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmin}} \left(g(\boldsymbol{x}) + \dfrac{\eta_t}{2} \|\boldsymbol{z}^t - \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\alpha}^t/\eta_t\|^2 \right) \\ \boldsymbol{z}^{t+1} = \underset{\boldsymbol{z}\in\mathbb{R}^m}{\operatorname{argmin}} \left(f(\boldsymbol{z}) + \dfrac{\eta_t}{2} \|\boldsymbol{z} - \boldsymbol{A}\boldsymbol{x}^{t+1} + \boldsymbol{\alpha}^t/\eta_t\|^2 \right) \\ \boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t + \eta_t(\boldsymbol{z}^{t+1} - \boldsymbol{A}\boldsymbol{x}^{t+1}). \end{cases}$$

- **z に関する最小化は** Prox **作用素** prox_f **（簡単）．**
- **x に関する最小化は行列 A が変数を絡ませるのでちょっと難しい．**
- 1 **反復あたりのコストが同じなら** FBS **より明らかに速い（理論的には不明）**
- **双対側での** Douglas Rachford Splitting **と等価** $\Rightarrow$ **ステップサイズ** η **によらず** ADMM **は安定．** (Lions & Mercier 76; Eckstein & Bertsekas 92)

テンソルの穴埋め問題への ADMM の適用

- **凸最適化の適用のポイント: テンソルの行列化** (Matricization)
- **テンソルが Tucker 分解の意味で低ランク**
  ⇔ そのテンソルの行列化は（行列の意味で）低ランク



テンソルの穴埋め問題への ADMM の適用

数学的な定式化:

$$\begin{aligned} \underset{\boldsymbol{x},\boldsymbol{z}_1,\ldots,\boldsymbol{z}_K\in\mathbb{R}^N}{\text{minimize}} \quad & \frac{1}{2\lambda}\|\boldsymbol{\Omega}\boldsymbol{x}-\boldsymbol{y}\|^2+\sum_{k=1}^{K}\gamma_k \underbrace{\|\boldsymbol{Z}_k\|_{\text{tr}}}_{\text{低ランク化}}, \\ \text{subject to} \quad & \boldsymbol{P}_k\boldsymbol{x}=\boldsymbol{z}_k \quad (k=1,\ldots,K), \end{aligned}$$

- $\boldsymbol{x}$ は推定すべきテンソルをベクトルとして書いたもの．
- $\boldsymbol{y}\in\mathbb{R}^M$ は観測．($M \ll N = n_1 n_2 \cdots n_K$)
- ベクトル化，行列化は要素の並び替え（線形変換）に過ぎない．
- ${\boldsymbol{P}_k}^\top \boldsymbol{P}_k = \boldsymbol{I}$（置換は直交変換）．
- すべてのモードが同時に低ランクになるように正則化．

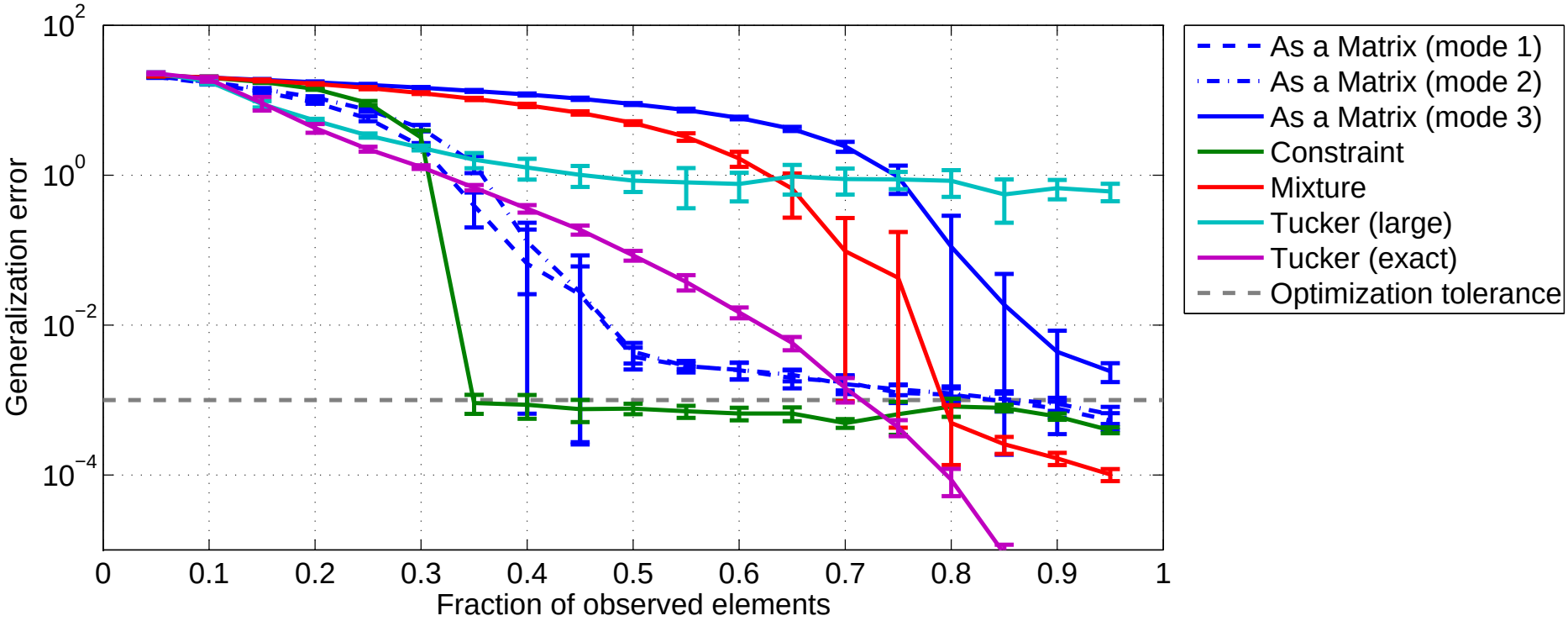
テンソルの穴埋め問題への ADMM の適用

拡張ラグランジアン

$$L_\eta(\boldsymbol{x}, \{\boldsymbol{Z}_k\}_{k=1}^K, \{\alpha_k\}_{k=1}^K) = \frac{1}{2\lambda}\|\Omega\boldsymbol{x} - \boldsymbol{y}\|^2 + \sum_{k=1}^K \gamma_k \|\boldsymbol{Z}_k\|_{\mathrm{tr}} + \sum_{k=1}^K \left(\alpha_k{}^\top (\boldsymbol{P}_k\boldsymbol{x} - \boldsymbol{z}_k) + \frac{\eta}{2}\|\boldsymbol{P}_k\boldsymbol{x} - \boldsymbol{z}_k\|^2 \right).$$

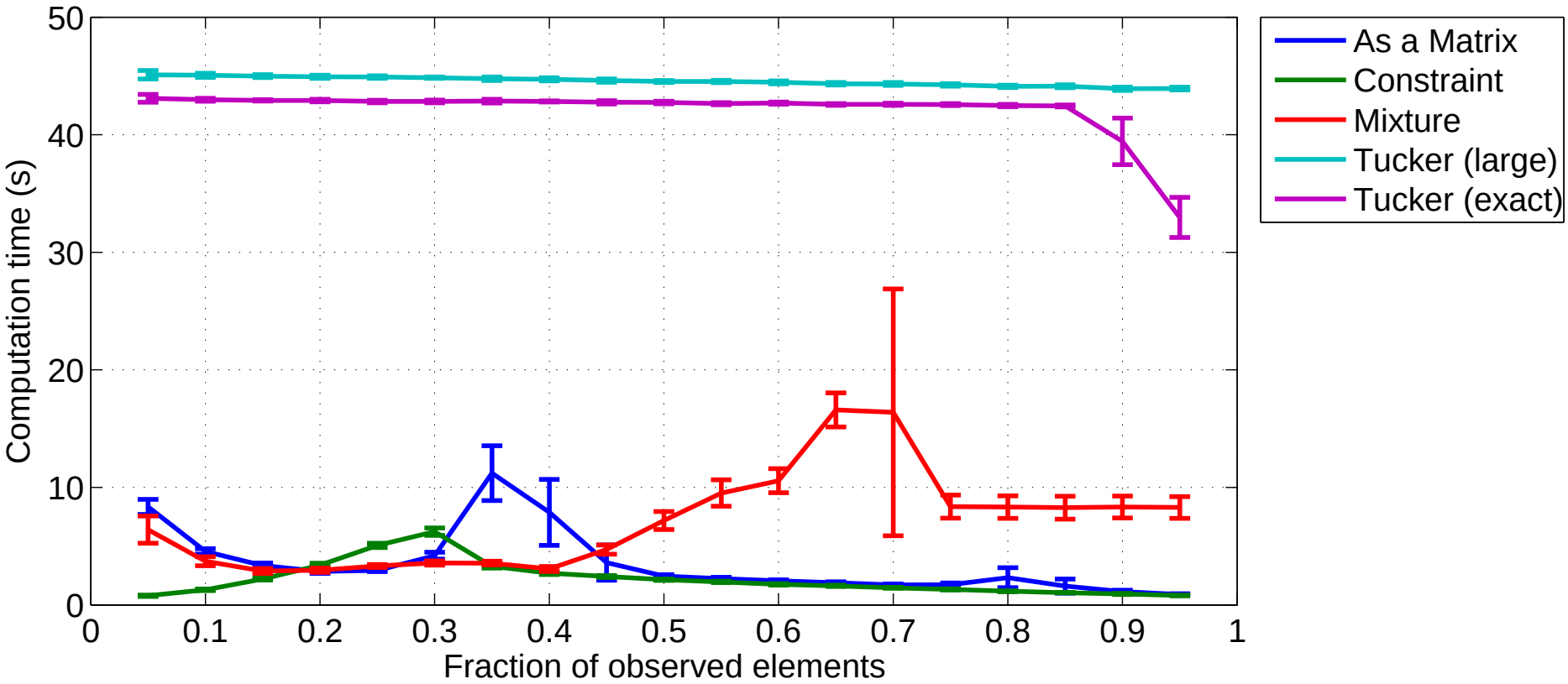
- $\boldsymbol{x}$ に関する最小化 $\boldsymbol{P}_k$ が直交行列なので解析的に $O(N)$ で計算可能.
- $\boldsymbol{Z}_k$（$\boldsymbol{z}_k$ を行列として並べたもの）に関する最小化はトレースノルムに関する Prox 作用素.
- ラグランジュ乗数ベクトルは制約の数（モードの数）だけ必要.

テンソル結果 1: 予測精度



- **提案手法** Constraint **は** 35% **くらい見えればほぼ完璧に予測可能．ランクを前もって決める必要なし．**
- **既存手法** Tucker(EM **アルゴリズム**) **はランクが合っていれば** OK**．ランクが間違っていると汎化誤差が収束しない．**

テンソル結果 2: 計算速度



- **しかも凸最適化は速い！** (Tomioka et al. 10)

Outline

凸最適化 $\Leftrightarrow$ 方程式のゼロ点

最小化問題

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x}).$$

方程式のゼロ点

Find $\boldsymbol{x}$ such that

$$(T_f + T_g)(\boldsymbol{x}) \ni \boldsymbol{0},$$

where

$$T_f = \partial f,$$

$$T_g = \partial g \quad (\textbf{劣微分作用素}).$$

例 (Prox 作用素):

$$\boldsymbol{x} = \mathrm{prox}_f(\boldsymbol{z}) = \operatorname*{argmin}_{\boldsymbol{x}' \in \mathbb{R}^n} \left(f(\boldsymbol{x}') + \frac{1}{2}\|\boldsymbol{x}' - \boldsymbol{z}\|^2 \right)$$

$$\Leftrightarrow \quad T_f(\boldsymbol{x}) + (\boldsymbol{x} - \boldsymbol{z}) \ni \boldsymbol{0}$$

$$\Leftrightarrow \quad \mathrm{prox}_f(\boldsymbol{z}) = (I + T_f)^{-1}(\boldsymbol{z})$$

凸最適化 $\Leftrightarrow$ 方程式のゼロ点

最小化問題

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x}).$$

方程式のゼロ点

Find $\boldsymbol{x}$ such that

$$(T_f + T_g)(\boldsymbol{x}) \ni \boldsymbol{0},$$

where

$$T_f = \partial f,$$

$$T_g = \partial g \quad (\textbf{劣微分作用素}).$$

例 (Prox **作用素**):

$$\boldsymbol{x} = \mathrm{prox}_f(\boldsymbol{z}) = \operatorname*{argmin}_{\boldsymbol{x}' \in \mathbb{R}^n} \left(f(\boldsymbol{x}') + \frac{1}{2}\|\boldsymbol{x}' - \boldsymbol{z}\|^2 \right)$$

$$\Leftrightarrow \quad T_f(\boldsymbol{x}) + (\boldsymbol{x} - \boldsymbol{z}) \ni \boldsymbol{0}$$

$$\Leftrightarrow \quad \mathrm{prox}_f(\boldsymbol{z}) = (I + T_f)^{-1}(\boldsymbol{z})$$

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} && f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

反復式:

$$\Downarrow$$

$$\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))$$

- Douglas Rachford Splitting **も同様の方法で導出できる（複雑すぎるので省略）**

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} && f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow\quad &\text{find } \boldsymbol{x} && \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

反復式:

$$
\Downarrow
$$

$$
\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))
$$

- Douglas Rachford Splitting **も同様の方法で導出できる（複雑すぎるので省略）**

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

反復式:

$$
\Downarrow
$$

$$
\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))
$$

- Douglas Rachford Splitting **も同様の方法で導出できる（複雑すぎるので省略）**

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

反復式:

$$
\Downarrow
$$

$$
\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))
$$

- Douglas Rachford Splitting **も同様の方法で導出できる（複雑すぎるので省略）**

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} && f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} && T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} && \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} && \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} && (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} && \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

$\Downarrow$

反復式:

$$\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))$$

- Douglas Rachford Splitting も同様の方法で導出できる（複雑すぎるので省略）

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

反復式:

$$
\Downarrow
$$

$$
\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))
$$

- Douglas Rachford Splitting も同様の方法で導出できる（複雑すぎるので省略）

Forward-Backward Splitting ($\Leftrightarrow$ IST) (Lions & Mercier 76)

$$
\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} \quad f(\boldsymbol{x}) + g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad T_f(\boldsymbol{x}) + T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \eta T_f(\boldsymbol{x}) + \eta T_g(\boldsymbol{x}) \ni \boldsymbol{0} \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} + \eta T_f(\boldsymbol{x}) \ni \boldsymbol{x} - \eta T_g(\boldsymbol{x}) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad (I + \eta T_f)(\boldsymbol{x}) = (\boldsymbol{x} - \eta T_g(\boldsymbol{x})) \\
\Leftrightarrow \quad &\text{find } \boldsymbol{x} \quad \boldsymbol{x} = \mathrm{prox}_{\eta f}(\boldsymbol{x} - \eta T_g(\boldsymbol{x}))
\end{aligned}
$$

反復式:

$$\Downarrow$$

$$\boldsymbol{x}^{t+1} = \mathrm{prox}_{\eta_t f}(\boldsymbol{x}^t - \eta_t \nabla g(\boldsymbol{x}^t))$$

- Douglas Rachford Splitting **も同様の方法で導出できる（複雑すぎるので省略）**

Outline

メッセージ

- ブラックボックス最適化から中身を考慮した最適化へ
- 微分不可能でも怖くない
- Prox 作用素や凸共役を計算しよう

Outline

凸共役の性質 (contd.)

- **凸共役（Legendre 変換）と畳み込み**:

$$(f+g)^*(\boldsymbol{y}) = \inf_{\boldsymbol{\alpha}} \left(f^*(\boldsymbol{y}-\boldsymbol{\alpha}) + g^*(\boldsymbol{\alpha})\right).$$

- **最小化と凸共役**:

$$\inf_{\boldsymbol{x}} f(\boldsymbol{x}) = -\sup_{\boldsymbol{x}} \left(\langle \boldsymbol{0}, \boldsymbol{x}\rangle - f(\boldsymbol{x})\right) = -f^*(\boldsymbol{0}).$$

- Fenchel **双対**:

$$\inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + g(\boldsymbol{x})\right) = -(f+g)^*(\boldsymbol{0}) = \sup_{\boldsymbol{\alpha}} \left(-f^*(-\boldsymbol{\alpha}) - g^*(\boldsymbol{\alpha})\right).$$

凸共役の性質 (contd.)

- **凸共役（Legendre 変換）と畳み込み**:

$$(f+g)^*(\boldsymbol{y}) = \inf_{\boldsymbol{\alpha}} \left(f^*(\boldsymbol{y}-\boldsymbol{\alpha}) + g^*(\boldsymbol{\alpha})\right).$$

- **最小化と凸共役**:

$$\inf_{\boldsymbol{x}} f(\boldsymbol{x}) = -\sup_{\boldsymbol{x}} \left(\langle \boldsymbol{0}, \boldsymbol{x}\rangle - f(\boldsymbol{x})\right) = -f^*(\boldsymbol{0}).$$

- Fenchel **双対**:

$$\inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + g(\boldsymbol{x})\right) = -(f+g)^*(\boldsymbol{0}) = \sup_{\boldsymbol{\alpha}} \left(-f^*(-\boldsymbol{\alpha}) - g^*(\boldsymbol{\alpha})\right).$$

凸共役の性質 (contd.)

- **凸共役（Legendre 変換）と畳み込み**:

$$(f+g)^*(\boldsymbol{y}) = \inf_{\boldsymbol{\alpha}} \left(f^*(\boldsymbol{y}-\boldsymbol{\alpha}) + g^*(\boldsymbol{\alpha})\right).$$

- **最小化と凸共役**:

$$\inf_{\boldsymbol{x}} f(\boldsymbol{x}) = -\sup_{\boldsymbol{x}} \left(\langle \boldsymbol{0}, \boldsymbol{x}\rangle - f(\boldsymbol{x})\right) = -f^*(\boldsymbol{0}).$$

- Fenchel **双対**:

$$\inf_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + g(\boldsymbol{x})\right) = -(f+g)^*(\boldsymbol{0}) = \sup_{\boldsymbol{\alpha}} \left(-f^*(-\boldsymbol{\alpha}) - g^*(\boldsymbol{\alpha})\right).$$

Fenchel 双対定理

$$\inf_{\boldsymbol{x}} \left(f(\boldsymbol{A}\boldsymbol{x}) + g(\boldsymbol{x})\right) = \sup_{\boldsymbol{\alpha}} \left(-f^*(-\boldsymbol{\alpha}) - g^*(\boldsymbol{A}^\top \boldsymbol{\alpha})\right).$$

- **等式制約付き最適化問題**

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n, \boldsymbol{z}\in\mathbb{R}^m}{\text{minimize}} \quad & f(\boldsymbol{z}) + g(\boldsymbol{x}), \\ \text{subject to} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{z} \end{aligned}$$

  の双対問題として導出できる．

  (http://www.ibis.t.u-tokyo.ac.jp/RyotaTomioka/Notes/DerivingDual)

- **凸共役の一覧表があれば機械的に双対問題を作れる点が美味しい．**

Fenchel 双対の例 (1): SVM

主問題

$$\min_{\boldsymbol{w}} \quad f(\boldsymbol{y} \circ \boldsymbol{X}\boldsymbol{w}) + \phi_\lambda(\boldsymbol{x})$$

$$\begin{cases} f(\boldsymbol{z}) = \sum_{i=1}^{m} \max(0, 1 - z_i), \\ \phi_\lambda(\boldsymbol{x}) = \frac{\lambda}{2}\|\boldsymbol{x}\|^2. \end{cases}$$

双対問題

$$\max_{\boldsymbol{\alpha}} \quad -f^*(-\boldsymbol{\alpha}) - \phi_\lambda^*(\boldsymbol{X}^\top(\boldsymbol{\alpha} \circ \boldsymbol{y}))$$

$$\begin{cases} f^*(-\boldsymbol{\alpha}) = \begin{cases} \sum_{i=1}^{m} -\alpha_i & (0 \le \boldsymbol{\alpha} \le 1), \\ +\infty & (\text{oterwise}), \end{cases} \\ \phi_\lambda^*(\boldsymbol{v}) = \frac{1}{2\lambda}\|\boldsymbol{v}\|^2. \end{cases}$$

Fenchel 双対の例 (2): 正則化付きロジスティック回帰 (Jaakkola & Haussler 99)

主問題

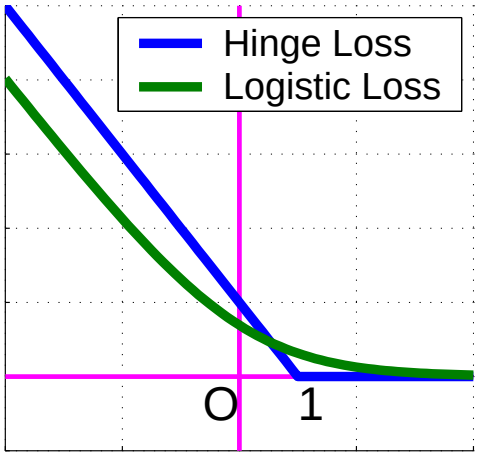
$$\min_{\boldsymbol{w}} \quad f(\boldsymbol{y} \circ \boldsymbol{X}\boldsymbol{w}) + \phi_\lambda(\boldsymbol{x})$$

$$\begin{cases} f(\boldsymbol{z}) = \sum_{i=1}^{m} \log(1 + \exp(-z_i)), \\ \phi_\lambda(\boldsymbol{x}) = \frac{\lambda}{2}\|\boldsymbol{x}\|^2. \end{cases}$$

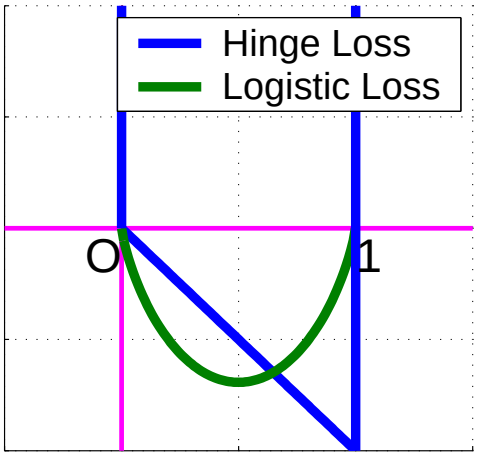
双対問題

$$\max_{\boldsymbol{\alpha}} \quad -f^*(-\boldsymbol{\alpha}) - \phi_\lambda^*(\boldsymbol{X}^\top(\boldsymbol{\alpha} \circ \boldsymbol{y}))$$

$$\begin{cases} f^*(-\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i \log(\alpha_i) \\ \qquad\qquad +(1-\alpha_i)\log(1-\alpha_i), \\ \phi_\lambda^*(\boldsymbol{v}) = \frac{1}{2\lambda}\|\boldsymbol{v}\|^2, \end{cases}$$



(a) primal losses



(b) dual losses

Fenchel 双対の例 (3): 線形計画

主問題

$$\begin{aligned} (\mathrm{P}) \quad \min \quad & \boldsymbol{c}^\top \boldsymbol{x}, \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq 0. \end{aligned}$$

$\Updownarrow$

主問題'

$$(\mathrm{P}') \quad \min_{\boldsymbol{x}} \quad f(\boldsymbol{A}\boldsymbol{x}) + g(\boldsymbol{x})$$

$$\begin{cases} f(\boldsymbol{z}) = \begin{cases} 0 & (\boldsymbol{z} = \boldsymbol{b}), \\ +\infty & (\text{otherwise}), \end{cases} \\ g(\boldsymbol{x}) = \begin{cases} \boldsymbol{c}^\top \boldsymbol{x} & (\boldsymbol{x} \geq 0), \\ +\infty & (\text{otehrwise}). \end{cases} \end{cases}$$

双対問題

$$\begin{aligned} (\mathrm{D}) \quad \max \quad & \boldsymbol{b}^\top \boldsymbol{y}, \\ \text{s.t.} \quad & \boldsymbol{A}^\top \boldsymbol{y} \leq \boldsymbol{c}. \end{aligned}$$

$\Updownarrow$

双対問題'

$$(\mathrm{D}') \quad \max_{\boldsymbol{y}} \quad -f(-\boldsymbol{y}) - g(\boldsymbol{A}^\top \boldsymbol{y})$$

$$\begin{cases} f^*(\boldsymbol{y}) = \boldsymbol{b}^\top \boldsymbol{y}, \\ g^*(\boldsymbol{\alpha}) = \begin{cases} 0 & (\boldsymbol{\alpha} \leq \boldsymbol{c}), \\ +\infty & (\text{oterwise}). \end{cases} \end{cases}$$

凸じゃない最適化は？

- **一般的に対処するのは困難**
- **基本的な方針は凸最適化を繰り返し解くこと**
  - EM
  - Difference of Convex (DC) programming
  - CCCP

文献

全体的なまとめ

- Tomioka, Suzuki, & Sugiyama (2011) Augmented Lagrangian Methods for Learning, Selecting, and Combining Features. In Sra, Nowozin, Wright., editors, *Optimization for Machine Learning*, MIT Press.
- Combettes & Pesquet (2010) Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer-Verlag.
- Boyd, Parikh, Peleato, & Eckstein (2010) Distributed optimization and statistical learning via the alternating direction method of multipliers.

教科書

- Rockafellar (1970) Convex Analysis. Princeton University Press.
- Bertsekas (1999) Nonlinear Programming. Athena Scientific.
- Nesterov (2003) Introductory Lectures on Convex Optimization: A Basic Course. Springer.

文献

Proximal Point Algorithm/Augmented Lagrangian

- Arrow, Hurwicz, & Uzawa (1958) Studies in Linear and Non-Linear Programming. Stanford University Press.
- Moreau (1965) Proximité et dualité dans un espace Hilbertien. Bul letin de la S. M. F.
- Rockafellar (1976) Monotone operators and the proximal point algorithm. SIAM J Control Optim 14, 877–898.
- Bertsekas (1982) Constrained Optimization and Lagrange Multiplier Methods. Academic Press.
- Tomioka, Suzuki, & Sugiyama (2011) Super-Linear Convergence of Dual Augmented Lagrangian Algorithm for Sparse Learning. Arxiv:0911.4046.

IST/FISTA

- Nesterov (2007) Gradient Methods for Minimizing Composite Objective Function.
- Beck & Teboulle (2009) A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. SIAM J Imag Sci 2, 183–202.

文献

Operator Splitting

- Lions & Mercier (1976) Splitting Algorithms for the Sum of Two Nonlinear Operators. SIAM J Numer Anal 16, 964–979.
- Gabay & Mercier (1976) A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Comput Math Appl 2, 17–40.
- Eckstein & Bertsekas (1992) On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators.

マルチタスク／マルチカーネル

- Evgeniou, Micchelli, & Pontil (2005) Learning Multiple Tasks with Kernel Methods. JMLR 6, 615–637.
- Bach, Thibaux, & Jordan (2005) Computing regularization paths for learning multiple kernels. Advances in NIPS, 73–80.
- Suzuki & Tomioka (2009) SpicyMKL. Arxiv:0909.5026.

文献

行列／テンソル

- Srebro, Rennie, & Jaakkola (2005) Maximum-Margin Matrix Factorization. Advances in NIPS 17, 1329–1336.
- Cai, Candès, & Shen (2008) A singular value thresholding algorithm for matrix completion.
- Tomioka, Suzuki, Sugiyama, & Kashima (2010) A Fast Augmented Lagrangian Algorithm for Learning Low-Rank Matrices. In ICML 2010.
- Tomioka, Hayashi, & Kashima (2010) On the extension of trace norm to tensors. ArXiv:1010.0789.
- Mazumder, Hastie, & Tibshirani (2010) Spectral Regularization Algorithms for Learning Large Incomplete Matrices. JMLR 11, 2287–2322.