THE UNIVERSITY OF TOKYO

# 数理情報工学特論第一
# 【機械学習とデータマイニング】
# 3章：分類④

かしま　ひさし
鹿島 久嗣
（数理 6 研）
kashima@mist.i.～

東京大学
THE UNIVERSITY OF TOKYO

DEPARTMENT OF MATHEMATICAL INFORMATICS

# 分類における$L_1$正則化と、データ間の関係の予測について学びます

- 分類における$L_1$正則化
- データ間の関係の予測
  - データの組に対する予測
  - 行列パラメータをもつモデル
  - トレースノルム正則化を用いた学習

# 分類における$L_1$正則化

# 分類の場合でも$L_1$正則化は有効です

- 回帰の場合と同じく、分類の場合にも、$L_1$正則化によって、疎なパラメータを得たい場合が多々ある
  - パラメータ$\mathbf{w}$の1-ノルム：

$$\| \mathbf{w} \|_1 = |w_1| + |w_2| + \cdots + |w_D|$$

  をペナルティ項として小さくすることで、多くの$w_d$が0になる効果

- 特に、データ数と比較して、特徴ベクトルの次元が高いような場合
  - 文書分類：文書中に出現する単語を用いた特徴ベクトル（bag-of-words）表現がしばしば用いられる
  - マイクロアレイ診断：各遺伝子の発現量を、患者の特徴ベクトルとして用いる
- これは数千～数十万次元にもなりうるため、大幅な次元削減が必要になることがある

# 分類の場合は、パラメータの最適解を閉じた形で得るのは難しいのでパラメータ徐々に改善する方式をとります

- $L_1$正則化を用いた場合の目的関数：

$$L(\mathbf{w}) \equiv \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\boldsymbol{\phi}(x^{(i)}); \mathbf{w}) - \lambda \parallel \mathbf{w} \parallel_1$$

- 分類問題の場合、この解を閉じた形で得るのは困難であるので、パラメータ$\mathbf{w}^{(t)}$から$\mathbf{w}^{(t+1)}$への更新によって、解を徐々に改善していく

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \boldsymbol{\Delta}^{(t)}$$

- 更新量$\boldsymbol{\Delta}^{(t)}$は目的関数$L$をなるべく大きくするように決定する

$$L(\mathbf{w}^{(t)} + \boldsymbol{\Delta}) > L(\mathbf{w}^{(t)})$$

# 目的関数を対数尤度の項と正則化の項にわけて考えます

- 目的関数：

$$L(\mathbf{w}) \equiv \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\phi(x^{(i)});\mathbf{w}) - \lambda \parallel \mathbf{w} \parallel_1$$

において、対数尤度の和の部分を

$$J(\mathbf{w}) \equiv \frac{1}{N}\sum_{i=1}^{N} \log P(y^{(i)}|\phi(x^{(i)});\mathbf{w})$$

とおき、目的関数を対数尤度の和と正則化に分け

$$L(\mathbf{w}) \equiv J(\mathbf{w}) - \lambda \parallel \mathbf{w} \parallel_1$$

と書くことにする

- 目指すのは以下を満たす$\boldsymbol{\Delta}$：

$$J(\mathbf{w}^{(t)}+\boldsymbol{\Delta})-\lambda \parallel \mathbf{w}^{(t)}+\boldsymbol{\Delta} \parallel_1 > J(\mathbf{w}^{(t)})-\lambda \parallel \mathbf{w}^{(t)} \parallel_1$$

# 対数尤度の項をテーラー展開します

- $L(\mathbf{w}^{(t)}+\boldsymbol{\Delta}) = J(\mathbf{w}^{(t)}+\boldsymbol{\Delta}) - \lambda\|\mathbf{w}^{(t)}+\boldsymbol{\Delta}\|_1$ を評価したい
- 対数尤度の項 $J(\mathbf{w})$を、現在のパラメータ$\mathbf{w}^{(t)}$の周りで2次までのテーラー展開により近似してみると：

$$L(\mathbf{w}^{(t)} + \boldsymbol{\Delta}) \approx$$

$$J(\mathbf{w}^{(t)}) + \boldsymbol{\Delta}^\top \boldsymbol{\nabla}(\mathbf{w}^{(t)}) + \frac{1}{2}\boldsymbol{\Delta}^\top \mathbf{H}(\mathbf{w}^{(t)})\boldsymbol{\Delta} - \lambda \parallel \mathbf{w}^{(t)} + \boldsymbol{\Delta} \parallel_1$$

  - $\boldsymbol{\nabla}(\mathbf{w}^{(t)})$： $J(\mathbf{w})$ の勾配
  - $\mathbf{H}(\mathbf{w}^{(t)})$： $J(\mathbf{w})$ のヘッセ行列

$$\boldsymbol{\nabla}(\mathbf{w}^{(t)}) \equiv \left( \left.\frac{\partial J(\mathbf{w})}{\partial w_1}\right|_{w=w_1^{(t)}}, \left.\frac{\partial J(\mathbf{w})}{\partial w_2}\right|_{w=w_2^{(t)}}, \cdots, \left.\frac{\partial J(\mathbf{w})}{\partial w_D}\right|_{w=w_D^{(t)}} \right)^\top$$

# ヘッセ行列を単位行列で置き換えることで、最急勾配法

- ヘッセ行列 $\mathbf{H}(\mathbf{w}^{(t)})$ の代わりに$\mathbf{H}(\mathbf{w}^{(t)}) \equiv -1/\eta^{(t)}\mathbf{I}$と置き換えると、

$$L(\mathbf{w}^{(t)}+\boldsymbol{\Delta}) \approx J(\mathbf{w}^{(t)})+\boldsymbol{\Delta}^{\top}\nabla(\mathbf{w}^{(t)})-\frac{1}{2\eta^{(t)}}\boldsymbol{\Delta}^{\top}\boldsymbol{\Delta}-\lambda \parallel \mathbf{w}^{(t)}+\boldsymbol{\Delta} \parallel_1$$

  - $\eta^{(t)}$：適当な正の定数（学習率）

- この近似は、$J(\mathbf{w}^{(t+1)})$ を一次近似するが、
  その近似精度は $\mathbf{w}^{(t+1)}$ が$\mathbf{w}^{(t)}$から離れるにつれ悪くなる

- なるべく現在の$\mathbf{w}^{(t)}$から離れないようにしようと働く項：
  $-1/2\eta^{(t)} \cdot \boldsymbol{\Delta}^{\top}\boldsymbol{\Delta} = -1/2\eta^{(t)} \|\boldsymbol{\Delta}\|_2^2$（すなわち、更新量の2ノルム）
  を、ペナルティ項として入れていると解釈することもできる

# 目的関数の最適化は次元ごとに行うことができます

- 目的関数は以下のように書くこともできる：

$$L(\mathbf{w}^{(t)}+\boldsymbol{\Delta}) \approx J(\mathbf{w}^{(t)})+\sum_{d=1}^{D}\left(\nabla_d(\mathbf{w}^{(t)})\Delta_d-\frac{\eta^{(t)}}{2}\Delta_d^2-\lambda|w_d^{(t)}+\Delta_d|\right)$$

- $L_1$回帰のときと同じように、各次元 $d$ ごとに分けて最大化を考えることができる
- つまり、目的関数を$\boldsymbol{\Delta}$ の各次元$\Delta_d$について最大化：

$$\Delta_d^{(t)} \equiv \underset{\Delta_d}{\text{argmax}}\ \nabla_d(\mathbf{w}^{(t)})\Delta_d-\frac{1}{2\eta^{(t)}}\Delta_d^2-\lambda|w_d^{(t)}+\Delta_d|$$

を行うことで最適な更新量$\Delta_d^{(t)}$ を各次元独立に得ればよい

# 整理すると、$L_1$正則化回帰のときに出てきた形式が現れます

- 次元ごとの最適化問題：

$$\Delta_d^{(t)} \equiv \underset{\Delta_d}{\text{argmax}} \ \nabla_d(\mathbf{w}^{(t)})\Delta_d - \frac{1}{2\eta^{(t)}}\Delta_d^2 - \lambda|w_d^{(t)} + \Delta_d|$$

- を整理し、定数部分を無視すると：

$$\Delta_d^{(t)} = \underset{\Delta_d}{\text{argmin}} \ \frac{\eta^{(t)}}{2}\left(\Delta_d - \eta^{(t)}\nabla_d(\mathbf{w}^{(t)})\right)^2 + \lambda|w_d^{(t)} + \Delta_d|$$

- さらにこれは更新式の定義 $\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \boldsymbol{\Delta}^{(t)}$ から：

$$w_d^{(t+1)} = \underset{w_d}{\text{argmin}} \ \frac{1}{2}\left(w_d - \left(w_d^{(t)} + \eta^{(t)}\nabla_d(\mathbf{w}^{(t)})\right)\right)^2 + \frac{\lambda}{\eta^{(t)}}|w_d|$$

  - どこかで見たことある形がでてくる

# $L_1$正則化回帰のときには場合分けで解が決まるのでした

- 目的関数：

$$w_d^{(t+1)} = \underset{w_d}{\text{argmin}} \ \frac{1}{2}\left(w_d - \left(w_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}^{(t)})\right)\right)^2 + \frac{\lambda}{\eta^{(t)}}|w_d|$$

は、ちょうど$L_1$回帰の次元ごとの解法を導いたとき出てきた形式であり、このときと同じ方法を用いることができる。

- $L_1$回帰における次元ごとの目的関数を思い出してみると：

$$w_d^* = \underset{w_d}{\text{argmin}} \ \frac{1}{2}(w_d - \tilde{w}_d)^2 + \gamma|w_d|$$

- このときの解は：
  - $\tilde{w}_d > \gamma$ ならば解は $w_d^* = \tilde{w}_d - \gamma$
  - $\tilde{w}_d < -\gamma$ ならば解は $w_d^* = \tilde{w}_d + \gamma$
  - $-\gamma \leq \tilde{w}_d \leq \gamma$ ならば $w_d^* = 0$

# 我々の場合（$L_1$正則化分類）も場合わけで解が求まります

- 2つの目的関数の間の対応関係：
  - $\tilde{w}_d = w_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}^{(t)})$
  - $\gamma = \lambda / \eta^{(t)}$
- 我々の問題の解は：

$$w_d^{(t+1)} = \begin{cases} w_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}^{(t)}) - \frac{\lambda}{\eta^{(t)}} & (\text{if } w_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}^{(t)}) > \frac{\lambda}{\eta^{(t)}}) \\ w_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}^{(t)}) + \frac{\lambda}{\eta^{(t)}} & (\text{if } w_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}^{(t)}) < -\frac{\lambda}{\eta^{(t)}}) \\ 0 & (\text{otherwise}) \end{cases}$$

# $L_1$正則化分類のアルゴリズムは、最急勾配法と（ある種の）丸め操作の繰り返しと見ることができます

- 前述の場合分けは、ある種の丸め操作であり、$\mathbf{w}_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}_d^{(t)})$ が0に近いときには解は0に丸められ、そうでない場合にも一定量だけ0に向かって引き戻されるように働く
- この場合分けの条件に入っている $\mathbf{w}_d^{(t)} + \eta^{(t)} \nabla_d(\mathbf{w}_d^{(t)})$ は、まさに最急勾配法の更新式と同じ形をしている
- 従って、アルゴリズムとしては：
  - 最急勾配法によってパラメータを更新したあとに
  - そのパラメータの値に応じて丸め操作を行う

  という2段階のパラメータ更新を繰り返し行っていると解釈できる

# $L_1$正則化分類のアルゴリズム

- 以下の2ステップを繰りかえす

1. 最急勾配法を適用し、中間的な解 $\tilde{w}_d$ を得る。

$$\tilde{\mathbf{w}} \equiv \mathbf{w}^{(t)} + \eta^{(t)} \nabla(\mathbf{w}^{(t)})$$

1. 以下の丸め操作によって新しいパラメータ $\mathbf{w}^{(t+1)}$を得る

$$w_d^{(t+1)} = \begin{cases} \tilde{w}_d - \frac{\lambda}{\eta^{(t)}} & (\text{if } \tilde{w}_d > \frac{\lambda}{\eta^{(t)}}) \\ \tilde{w}_d + \frac{\lambda}{\eta^{(t)}} & (\text{if } \tilde{w}_d < -\frac{\lambda}{\eta^{(t)}}) \\ 0 & (\text{otherwise}) \end{cases}$$

# データの組に対する予測

# 分類問題の一般化として、2つ（複数）の入力をもつ分類問題を考えます

- これまでは入力$x$に対して出力$y$を予測するという一対一の関係を扱っていた
- 時によって2つの入力$x$と$x'$の組（より一般的には$m$個の入力）に対して出力$y$を予測したい場合がある
- 例：ネットワーク構造の予測問題
  - ネットワーク構造のリンク構造が部分的に与えられた（いくつかのノード対に関して、それらの間にリンクがあるかないかという情報が与えられた）ときに、残りの部分についてのリンク構造の予測を行う問題
    - タンパク質の相互作用予測：2つのタンパク質（データ）の間に物理的な相互作用（チームで働くなど）があるかを予測
    - 購買予測：顧客と商品の間の購買関係を予測

## 2つの入力をもつ条件付き分布を推定する問題を考えます

- リンク予測の一番シンプルな捉え方：2つのノードの2クラス分類
  - 2つのノード$x$ と$x'$ に対し、それらの間にリンクが存在する（+1）かしないか（-1）をクラスラベルとする
- つまり、2つの入力データが与えられた時の、出力の条件付き分布$P(y|\, x, x')$を推定する
- $x$ と $x'$ は同じ集合$\mathcal{X}$ に属していても良いし、別々のノード集合に属して（$x$は集合$\mathcal{X}$ に、$x'$は集合$\mathcal{X}'$に属する）もよい
  - $\mathcal{X}$ を顧客の集合、$\mathcal{X}'$を商品の集合とすると、ある顧客 $x \in \mathcal{X}$ がある商品$x' \in \mathcal{X}'$を購入するかどうかを予測するというマーケティングの文脈での予測問題を考えることができる
- より一般的に、複数種類の関係がある場合（多クラス分類）や、数値的な関係がある場合（回帰）なども考えられる

# 2入力のロジスティック回帰モデルを考えます

- 入力の組 $(x, x')$ についての分類問題を解くために、入力の組が与えられた時の出力の条件付き確率$P(y|x, x')$をモデル化する
- ロジスティック回帰モデルを2入力に拡張したモデルを考える：

$$P(y = +1|x, x'; \mathbf{w}) \equiv \sigma(\mathbf{w}^\top \boldsymbol{\psi}(x, x'))$$

  - $\sigma(z) \equiv (1+\exp(-z))^{-1}$ ：シグモイド関数
  - $\boldsymbol{\psi}(x, x')$ ： 2つの入力 $x$と $x'$ の組み合わせに対する特徴ベクトル
  - $\mathbf{w}$：パラメータベクトル
- 通常のロジスティック回帰との違いは、特徴ベクトルが2つの入力の組み合わせに対して定義されているところ
- 2つの入力それぞれの特徴ベクトル$\boldsymbol{\phi}(x)$ および$\boldsymbol{\phi}(x')$ が予め与えられているものとして、これらを利用して$\boldsymbol{\psi}(x, x')$を設計することが重要

# 組み合わせ特徴ベクトルの定義として良く用いられるのは2つの入力それぞれの特徴の組み合わせを用いる表現です

- 2つの入力ペアに対する特徴ベクトル $\psi(x, x')$ をそれぞれの特徴ベクトル$\phi(x)$ および $\phi(x')$ の特徴の組み合わせで構成する
- つまり、$\phi(x)$ および $\phi(x')$ の次元がそれぞれ$D$および$D'$であるとするときに、これらの間の $D\,D'$ 個の組み合わせ特徴を定義する：

$$\psi(x, x') \equiv \phi(x) \otimes \phi(x')$$

  - $\otimes$ はクロネッカー積と呼ばれる演算子
  - 要素ごとに書けば $\psi_{(i\text{-}1)\mathrm{D}'+j}(x, x') \equiv \phi_i(x)\ \ \phi_j(x')$

# 一旦特徴ベクトルが定義できれば、あとはこれまでと同じように学習を行うことができます

- 訓練データ集合：
  - 各訓練データを2つの入力の組$(x^{(i)}, x'^{(i)})$と対応する出力$y^{(i)}$とする
  - これを$N$組集めたもの$\{ (x^{(i)}, x'^{(i)}, y^{(i)})\}_{i=1}^{N}$
- 例えば、2-ノルムに基づく正則化（もしくは正規分布を事前分布とする事後確率最大化）を用いたとすれば、最適化問題：

$$\mathbf{w}^* \equiv \underset{\mathbf{w}}{\mathrm{argmax}} \ \frac{1}{N} \sum_{i=1}^{N} P(y^{(i)} | x^{(i)}, x'^{(i)}; \mathbf{w}) - \lambda \parallel \mathbf{w} \parallel_2^2$$

  を解けば、最適なパラメータ$\mathbf{w}^*$が求まる
- このモデルでは必然的に特徴ベクトルの次元は$D\,D'$と高くなってしまうため、パラメータを疎にするために$\mathrm{L}_1$正則化を用いるのもよい

# 行列パラメータをもつモデル

# 2入力のロジスティック回帰は行列を用いて表現できます

- 2つの入力の組に対する特徴ベクトルは、個々の入力の特徴の組み合わせで作られることから、特徴ベクトルを行列で自然に表現することができる
- つまり、$D\,D'$次元の特徴ベクトル$\boldsymbol{\psi}(x, x') \equiv \boldsymbol{\phi}(x) \otimes \boldsymbol{\phi}(x')$の定義の代わりに、$D \times D'$の特徴行列$\boldsymbol{\Psi}(x, x')$を次のように定義する：

$$\boldsymbol{\Psi}(x, x') \equiv \boldsymbol{\phi}(x) \otimes \boldsymbol{\phi}(x')^\top$$

- これに対応して、パラメータのほうも$D \times D'$の行列$\mathbf{W}$として書くことにすると、ロジスティック回帰モデルは：

$$P(y = +1 | x, x'; \mathbf{W}) \equiv \sigma(\mathrm{Tr}\mathbf{W}^\top \boldsymbol{\Psi}(x, x'))$$

  - Tr は行列のトレースを表す
  - 上式におけるTrの中身はちょうど$\mathbf{W}$と$\boldsymbol{\Psi}(x, x')$の要素ごとの積の和

# 学習の目的関数も同様に行列を用いて書き直すことができます

- 学習の目的関数も同じように書き直すことができる
- 例えば、2-ノルムに基づく正則化の場合には、行列表現を使うと：

$$\mathbf{W}^* \equiv \underset{\mathbf{W}}{\text{argmax}} \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}, x'^{(i)}; \mathbf{W}) - \lambda \| \mathbf{W} \|_F^2$$

  - $\| \cdot \|_F^2$ はフロベニウスノルム（行列の全ての要素の2乗和）
    - これは行列をベクトルだと思って2-ノルムをとったときに等しい

# 入力とパラメータを行列だとみなすことによって、新たな方向性が見えきます

- 入力行列は必ずしもベクトルのクロネッカー積による定義のようにランク1の行列である必要はなく、一般の$D \times D'$の行列$\boldsymbol{\Psi}(x, x')$であるとしても問題ない
  - 画像は各要素が色の濃淡等を表す行列で自然に表現できる
  - 脳波解析などにおいて、複数の時系列に対する分類を行いたい場合、特徴の定義として、時系列間の相関係数を（対称）行列として表現することがある
- しかし、モデルを行列で表現したところで、これはベクトル表現による以前のモデルと等価であり、これらに伴う最適化問題もまた等価であるため、表現の違いによる本質的な変化はない
- 入力が行列という構造をもつことが意味をもつためには、モデルの学習において行列の構造を明示的に利用する必要がある

# 「行列であること」を明示的に扱うためには 行列の複雑さを正則化項に考慮する必要があります

- 入力が行列という構造をもつことが意味をもつためには、モデルの学習において行列の構造を明示的に利用する必要がある
- その行列構造をどこに入れるか？ → 正則化項に入れる
- 最適化問題における正則化項であるフロベニウスノルム（2-ノルム）の代わりに、行列の複雑さを表す別の指標を考えることで、学習において行列の構造を明示的に考慮する

$$\mathbf{W}^* \equiv \underset{\mathbf{W}}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}, x'^{(i)}; \mathbf{W}) - \lambda \| \mathbf{W} \|_F^2$$

⇩

？

# 行列パラメータの複雑さを測るひとつの基準は、行列のランク（階数）です

- 行列の複雑さの1つの指標として行列のランク（階数）が考えられる
- **W**のランク：**W** の固有値（$\min\{D, D'\}$個ある）を $\boldsymbol{\mu} \equiv (\mu_1(\mathbf{W}), \mu_2(\mathbf{W}), \ldots, \mu_{\min\{D, D'\}}(\mathbf{W})$ としたときの、$\boldsymbol{\mu}$ の非零要素の数
- ランクが小さいこと、つまり、固有値の集合における非零要素が少ないことが、行列の複雑さが低いことを表す
  - $\boldsymbol{\mu}$の非零要素だけを取り出して並べたベクトルを$\boldsymbol{\mu}_+$と書くことにすると、**W** は以下のように分解できることが知られている：

$$\mathbf{W} = \mathbf{U}\mathrm{diag}(\boldsymbol{\mu}_+)\mathbf{V}^\top$$

  - **diag**($\boldsymbol{\mu}_+$) は $\boldsymbol{\mu}_+$を対角成分としてもつような対角行列
  - $\boldsymbol{\mu}_+$の非零要素の数を$R$とすると、**U**と**V**はそれぞれ $D \times R$および $D' \times R$の行列であり、ランク$R$が小さいほど、**W**を小さな行列（**U**と**V**）で表現できることがわかる

# ランクを落とすための正則化項として、行列の固有値の1-ノルムを用います

- 行列のフロベニウスノルムに代わる正則化項として、行列のランク$R(\mathbf{W})$を用いるのがよいという気がしてくるが、残念ながら$R(\mathbf{W})$は凸関数ではなく、最適化の観点からは正則化項に不向きである
- そこで、ランクの代わりに固有値ベクトル$\boldsymbol{\mu}$の1-ノルムを用いることを考える（これは凸関数であることが知られている）
- そこで、$\mathbf{W}$の固有値ベクトル$\boldsymbol{\mu}(\mathbf{W})$に対する$L_1$正則化項を入れる：

$$\mathbf{W}^* \equiv \underset{\mathbf{W}}{\text{argmax}} \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}, x'^{(i)}; \mathbf{W}) - \lambda \| \boldsymbol{\mu}(\mathbf{W}) \|_1$$

- $L_1$正則化は結果として得られるパラメータの多くを0にする効果があるため、これを固有値に対して用いることによって、多くの固有値を0にし、結果として行列のランクを低くする

# 固有値の1-ノルムを用いた正則化は、近年、盛んに利用されつつあります

- 行列の固有値ベクトルに対する1-ノルムはトレースノルムや核(nuclear)ノルムと呼ばれ、これを用いた正則化は、トレースノルム正則化、スペクトル正則化などと呼ばれる
- 協調フィルタリングやネットワーク予測、マルチタスク学習など行列がパラメータとして現れるに様々な場面において、近年盛んに用いられている

# トレースノルム正則化を用いた学習

# トレースノルム正則化を用いた学習の最適化問題は$L_1$正則化のときと同じように2段階更新で行います

- 目的関数の最適化：

$$\mathbf{W}^* \equiv \underset{\mathbf{W}}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)}|x^{(i)}, x'^{(i)}; \mathbf{W}) - \lambda \| \boldsymbol{\mu}(\mathbf{W}) \|_1$$

- 例によって、現在のパラメータ$\mathbf{W}^{(t)}$から$\mathbf{W}^{(t+1)}$への更新によって、解を徐々に改善していくことにする
- $L_1$正則化のときと同じように：

1. パラメータ$\mathbf{W}=\mathbf{W}^{(t)}$における最急勾配による更新によって中間的なパラメータ$\tilde{\mathbf{W}}$を一旦得る
2. 丸め操作によって最終的な更新パラメータ$\mathbf{W}^{(t+1)}$を得る

という2段階の更新を繰り返す

# ステップ1:（対数尤度部分についての）最急勾配法

- まず、最急勾配法によって、中間的なパラメータ $\tilde{\mathbf{W}}$ を計算する

$$\tilde{\mathbf{W}} \equiv \mathbf{W}^{(t)} + \eta^{(t)} \nabla(\mathbf{W}^{(t)})$$

  - $\nabla(\mathbf{W}^{(t)})$は、目的関数の対数尤度部分：

$$\frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)} | x^{(i)}, x'^{(i)}; \mathbf{W})$$

のW=W(t)における勾配（行列）であり、その$(k,l)$ 要素の定義は：

$$[\nabla(\mathbf{W}^{(t)})]_{k,\ell} \equiv \frac{\partial}{\partial [\mathbf{W}]_{k,\ell}} \frac{1}{N} \sum_{i=1}^{N} \log P(y^{(i)} | x^{(i)}, x'^{(i)}; \mathbf{W}) \Big|_{[\mathbf{W}]_{k,\ell} = [\mathbf{W}^{(t)}]_{k,\ell}}$$

# ステップ2: ステップ1で得られた $\tilde{\mathbf{W}}$ の固有値の丸め操作によって更新パラメータ $\mathbf{W}^{(t+1)}$ を得ます

- $L_1$正則化のときには、中間パラメータに丸め操作を行うことで最終的なパラメータを得たが、今回の正則化項 $\|\boldsymbol{\mu}(\mathbf{W})\|_1$ は固有値に対する$L_1$ノルムであるため、$\tilde{\mathbf{W}}$ の固有値に対して丸め操作を行う
- まず$\tilde{\mathbf{W}}$ の特異値分解 $\tilde{\mathbf{W}} = \mathbf{U}\mathbf{diag}(\boldsymbol{\mu}(\tilde{\mathbf{W}}))\mathbf{V}^\top$ を行う
- $\boldsymbol{\mu}(\mathbf{W})$の各要素 $\mu_j(\mathbf{W})$ ($j$=1,2,…,min{$D$, $D'$})に対して、$L_1$正則化のときの丸め操作と同様の操作によって $\tilde{\mu}_j$ を得る：

$$\tilde{\mu}_j = \begin{cases} \mu_j(\tilde{\mathbf{W}}) - \frac{\lambda}{\eta^{(t)}} & (\text{if } \mu_j(\tilde{\mathbf{W}}) > \frac{\lambda}{\eta^{(t)}}) \\ \mu_j(\tilde{\mathbf{W}}) + \frac{\lambda}{\eta^{(t)}} & (\text{if } \mu_j(\tilde{\mathbf{W}}) < -\frac{\lambda}{\eta^{(t)}}) \\ 0 & (\text{otherwise}) \end{cases}$$

⇨ 纏めてベクトル $\tilde{\boldsymbol{\mu}}$ として書く

- 疎になった $\tilde{\boldsymbol{\mu}}$ を用いて$\mathbf{W}^{(t+1)}$ を得る：

$$\mathbf{W}^{(t+1)} = \mathbf{U}\mathbf{diag}(\tilde{\boldsymbol{\mu}}(\tilde{\mathbf{W}}))\mathbf{V}^\top$$