複合情報学特別講義第二

拡がる機械学習の可能性
— グラフとネットワークの機械学習を中心として

東京大学大学院
情報理工学系研究科
鹿島 久嗣
東京大学
DEPARTMENT OF MATHEMATICAL INFORMATICS

機械学習による関係データの解析手法について紹介します

- 関係データとは
  - 関係データとは何か
  - 関係データにはどのようなものがあるか
  - 関係データ解析のタスク
- 関係データの表現
  - 行列と多次元配列
- 関係データの解析方法
  - 低ランク性の仮定
  - 特異値分解
  - テンソル分解
- 関係データ解析の応用

関係データ

近年、データ間の関係の解析が注目を浴びつつある

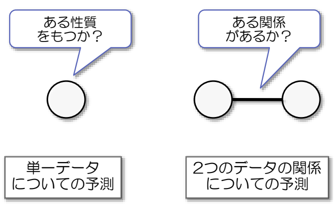
- 従来：「個々のデータを対象とした解析」

  ⇩

  近年：「データの間の関係の解析」
- データ間の関係に注目することで、
  個々のデータに注目しているだけでは見えない性質が見えてくる
  - コンピュータネットワーク上のプロセス依存関係から異常を予測
  - 複数の脳波時系列の相関関係から思考を読みとる
- 関係の分析は様々な領域において盛んに行われつつある
  - ソーシャルネットワーク分析：人間関係
  - オンラインショッピング：顧客と商品の間の関係

関係データとは ものごとの関係を表現したデータ

- 通常のデータ解析では、ひとつのデータについて成り立つ性質を推論する
- 関係データとは：データの組についてのデータ
- 関係の成立や、関係のもつ性質についての推論を行う



関係データの例：マーケティング、Web、バイオ、…

- オンラインマーケティング
  - 顧客と商品との間の関係（購買、評価）
- ソーシャルネットワーク
  - SNS内の人間関係 (facebook, twitter, mixi, …)
  - 企業間取引
- 生体ネットワーク
  - タンパク質相互作用ネットワーク
  - 化合物-タンパク質相互作用

関係データを用いたタスク：予測と発見

- 予測
  - 推薦システム（協調フィルタリング）
    - 顧客と商品との間の関係（購買、評価）を予測
      - 例： Netflix challenge
  - SNSの友人推薦
  - 新規薬剤候補の探索
- 発見
  - 顧客セグメンテーションの発見
  - 協調するタンパク質グループの発見
  - 例外の発見

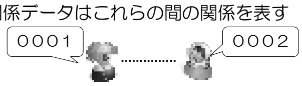
関係データの形式的表現

関係データの表現：グラフや行列など

- 通常、データは表形式で与えられる

顧客番号	顧客氏名	年齢	性別	住所	...
0001	○○	40代	男性	東京都	...
0002	××	30代	女性	大阪府	...

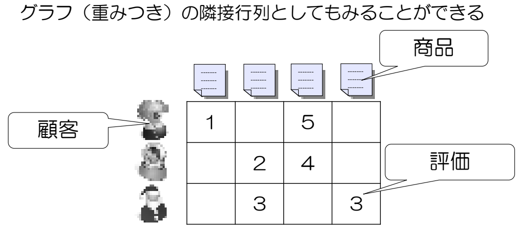
- 関係データはこれらの間の関係を表す



- 数学的な表現
  - 行列／多次元配列
  - グラフ／ハイパーグラフ

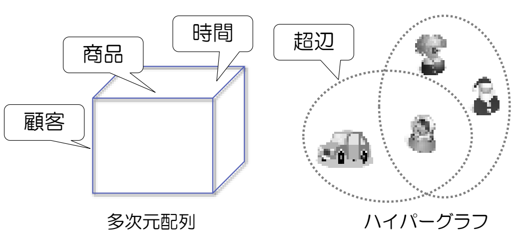
2項関係の集合は行列として表現できる

- 2項関係は行列として表現できる
  - 行と列がデータの集合に対応
  - 各要素がデータ間の関係を表す
- グラフ（重みつき）の隣接行列としてもみることができる



多項関係の集合は多次元配列やハイパーグラフとして表現できる

- 多項関係の集合は多次元配列として表現できる
- ハイパーグラフとしても表現可能
  - こちらのほうがより一般的：関係に参加するデータの数が可変



行列データ（２項関係）の解析手法

行列データの解析手法

- 行列の補完問題を考える
- 協調フィルタリングの初等的手法：GroupLens
- 行列の低ランク分解
- トレースノルム

GroupLens：協調フィルタリングの初等的手法

- 推薦システム（協調フィルタリング）は、顧客と商品との間の関係（購買、評価）を予測する
- 値が分かっている部分から、わかっていない部分を予測したい
- GroupLens：初期の予測アルゴリズム
  - ニュースの推薦が目的

商品

顧客

評価

1		5	
	2	4	
	3		3

GroupLensでは、ある顧客の評価を、似た顧客の評価を持ってきて予測する

- 予測したい顧客と似た顧客を集め、類似顧客の評価を用いて予測を行う
  - Aさんの未知要素を予測したいとする
  - Aさんと良く似た評価を行っている別の顧客を集めてきて、彼らの評価を用いて予測する

Aさんに似た人

Aさん

知りたい要素

1	2	5	5
	2	4	5？
	5		3

「似ている」の定義は 評価値の相関係数で測り、相関係数で重みづけして予測します

- 2人の顧客の類似度を（共に評価値が観測されている部分の）相関係数で測る
- 相関係数で重みづけし予測を行う

$$y_{i,j} = y_i + \Sigma_{k\neq i}\, \rho_{i,k} (y_{k,j} - y_k) \,/\, \Sigma_{k\neq i}\, \rho_{i,k}$$

- 同様に、商品間の類似度を用いることも可能

相関係数

相関係数

重みつき予測

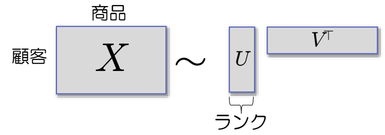
1	2	5	5
	2	4	4.5
	5		3

協調フィルタリングの初等的手法は行列の低ランク性を暗に仮定している

- 行列の各行が、別の行の（相関係数で重み付けた）線形和によって表せるとしている
  - 線形従属
- 対象となる行列のランクがフルランクではない（⇒低い）ことを暗に仮定した方法ということになる
- 低ランク性の仮定は行列の穴埋めに有効であろう
  - データよりもパラメータが多い状況では、なんらかの事前知識を用いて解に制約を設ける必要がある
  - 低ランク性の仮定は、実質パラメータ数を減らす

低ランク性を仮定する

- 低ランク性の仮定：行列が2つの（薄い）行列の積で書ける

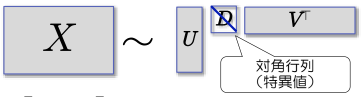


$$\text{minimize}_{Y}\ \|X - Y\|_F^2 \ \text{ s.t. } \operatorname{rank}(Y) \leq k$$

- 実効パラメータ数が減っている
- U（V）の各行：顧客（商品）の特徴を捉えた低次元の潜在空間にデータを配置
  - この空間で近いものが似た顧客（商品）：グループ構造

特異値分解もよく用いられる

- 行列分解 $X = UV^{\top}$ の仮定だけでは、解の不定性があるので、制約を入れる
- 特異値分解



  - 制約： $U^{\top}U = I \;\; V^{\top}V = I$
- $X^{\top}X$ の固有値問題になる
  - 固有値を大きい方からk個とる

欠損値がある場合には特異値分解は使えない

- ランク制約をもった最適化問題は凸最適化問題ではない
  - ランクk以下の行列は凸集合ではない
    - 目的関数 ＝ 復元誤差（凸関数）＋ランク制約

$$\mathrm{minimize}_{Y} \; \|X - Y\|_F^2 \;\; \text{s.t. } \mathrm{rank}(Y) \leq k$$

もしくは分解を $UV^{\top}$ と明示的におくと誤差項が非凸になってしまう

$$\mathrm{minimize}_{Y} \; \|X - UV^{\top}\|_F^2$$

- 全データが観測されている場合には、固有値問題としてたまたま解ける
- 欠損値がある場合には困る

欠損値がある場合には、EMアルゴリズムが用いられる

- ひとつの方法としては気にせず、勾配法などで適当に解く
  - データが大きいときにはこちら
- EMアルゴリズム：未観測部分には暫定的な推定値をあてはめ、完全観測として問題を解く
  1. 未観測部分を適当に初期化（平均など）
  2. 低ランク行列分解を適用
  3. 復元した値で未観測部分の値を置き換える

  ステップ 2~3を収束まで繰り返す

凸最適化としての定式化：トレースノルム正則化

- 行列のランク制約は凸集合ではないので、凸集合であり、ランク制約のよい近似となるような制約がほしい
- 行列の特異値の和を用いる　特異値

$$\|Y\|_* = \sigma_1(Y) + \sigma_2(Y) + \ldots$$

  - 特異値の集合（$\sigma_1(Y), \sigma_2(Y), \ldots$）に対する$L_1$ノルム制約と等価であるため、疎になる⇒ランクが落ちる
  - 一方、ランクは、非零の特異値の個数
- 目的関数 ＝ 観測部分の復元誤差 ＋ トレースノルム制約

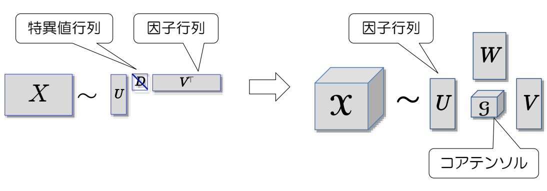
$$\mathrm{minimize}_{Y} \; \|O * (X - Y)\|_F^2 \;\; \text{s.t. } \|Y\|_* \leq c$$

  - 最適化は勾配法と特異値分解の組み合わせ

多次元配列（多項関係）の解析手法

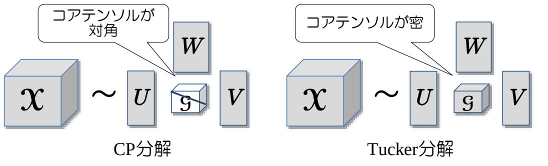
行列分解は多次元配列（テンソル）の低ランク分解に一般化される

- 行列の低ランク分解の多次元配列への一般化
  - ちいさな（コア）テンソルと因子行列に分解する
- 近年、機械学習やデータマイニングで盛んに用いられている



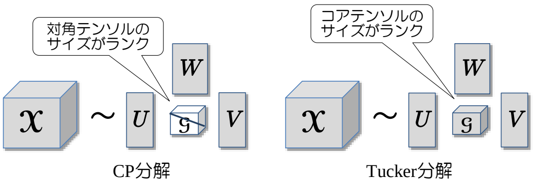
テンソル分解のタイプ：CP分解とTucker分解

- よく用いられるのがCP分解とTucker分解
- CP分解：特異値分解の自然な拡張（コアテンソルが対角；正方）
- Tucker分解：よりコンパクトな表現（みっちりコア；各モードの次数が異なる）



テンソルのランクは分解のタイプによって決まる

- 行列のランクはSVDの非零の特異値の数で決まった
- テンソル分解の場合には分解のタイプによって決まる
  - CP分解、Tucker分解それぞれでランクの定義がある

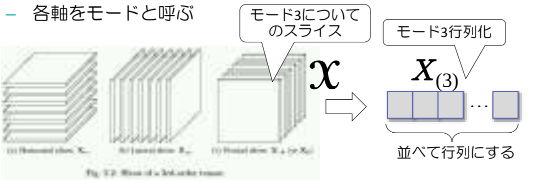


補足：テンソルの基本的演算

- いくつか特殊な操作が用いられる
  - 行列化：テンソルを行列に変換する操作
  - モード積：テンソルと行列の掛け算

補足：テンソルの行列化

- それぞれのモードに対してスライスが定義される
  - 各軸をモードと呼ぶ

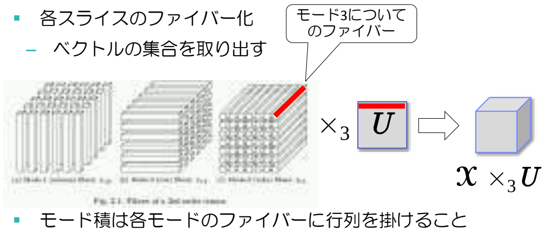


- 行列化はそれぞれのモードについてスライスをつなげたもの
  - $\mathcal{X}$ が3階テンソルなら、モードごとに$X_{(1)}, X_{(2)}, X_{(3)}$ の3つの行列ができる (4階以上も同様に定義される)

* The figures are taken from T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review, 51(3):455–500, 2009.*

補足：テンソルのモード積

- 各スライスのファイバー化
  - ベクトルの集合を取り出す

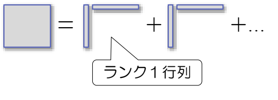


- モード積は各モードのファイバーに行列を掛けること
- $\mathcal{X} \times_3 U$ は 1-モード積の各ファイバーに行列U を掛ける
  - $\mathcal{Y} = \mathcal{X} \times_n U \Leftrightarrow Y_{(n)} = UX_{(n)}$

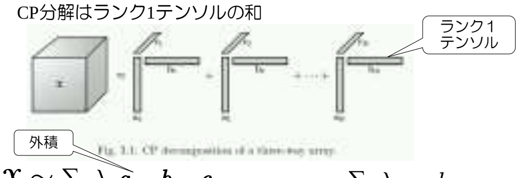
* The figures are taken from T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review, 51(3):455–500, 2009.*

CP分解はランク１テンソルの和として定義される

- 行列はランク1行列の和



- CP分解はランク1テンソルの和



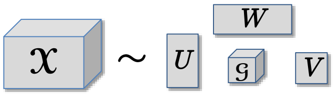
$\mathcal{X} \sim \sum_r \lambda_r \boldsymbol{a}_r \circ \boldsymbol{b}_r \circ \boldsymbol{c}_r$ $\quad x_{ijk} = \sum_r \lambda_r a_{ri} b_{rj} c_{rk}$

* The figures are taken from T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review, 51(3):455–500, 2009.*

Tucker分解は小さいテンソルと行列によって定義される

- Tucker分解はコアテンソルと、因子行列によって定義される
  - モード積を使って定義される

$$\mathcal{X} \sim \mathcal{G} \times_1 U \times_2 V \times_3 W \quad (x_{ijk} = \textstyle\sum_{pqr} g_{pqr}\, u_{ip}\, v_{iq}\, w_{ir})$$



  - 多くの場合因子行列の列ベクトルが正規直交であると仮定
- CP分解はコアテンソルが対角であるようなTuckerの特殊ケース

テンソル分解の方法：繰り返し最適化による準最適化

- 基本的には与えられたテンソルを2乗誤差の意味で最適近似するような分解を求める

$$\text{minimize} \ \ \|\mathcal{X} - \mathcal{Y}\|_F^2 \ \ \text{s.t.} \ \ \mathcal{Y} = \mathcal{G} \times_1 U \times_2 V \times_3 W$$

- 最適解を求めるのは難しい
  - 凸最適化ではない
  - 特異値分解などで都合よく解が求まったりしない
- 繰り返し最適化を行うのが一般的
  - 最小二乗回帰もしくは特異値分解の繰り返し

CP分解の方法：繰り返し最小二乗法（ALS）

- 解きたいのは

$$\text{minimize}_{\mathcal{Y}} \ \|\mathcal{X} - \mathcal{Y}\|_F^2 \ \ \text{s.t.} \ \ \mathcal{Y} = \mathcal{G} \times_1 U \underbrace{\times_2 V \times_3 W}_{=Z_{(1)}}$$

- Uについて最適化する（V, W についても同様）：

$$\mathcal{Y} = \mathcal{Z} \times_1 U \Leftrightarrow Y_{(1)} = UZ_{(1)}$$

  を使って目的関数を書き換えると

$$\|\mathcal{X} - \mathcal{Y}\|_F^2 = \|Y_{(1)} - UZ_{(1)}\|_F^2$$

  これは最小二乗回帰
- $\mathcal{G}$ は U, V, Wに吸収してよい（単位テンソル）
- 繰り返し最小二乗法（ALS）と呼ばれる

Tucker分解の方法：固有値問題の繰り返し

- Tucker分解は直交条件が加わる：

$$\text{minimize} \ \ \|\mathcal{X} - \mathcal{Y}\|_F^2 \ \ \text{s.t.} \ \ \mathcal{Y} = \mathcal{G} \times_1 U \times_2 V \times_3 W$$

$$\text{s.t.} \ U^\top U = I, \ V^\top V = I, \ W^\top W = I$$

  - Uについて最適化する（V, W についても同様）：
    - $\max_U \|\mathcal{X} - \mathcal{Y}\|_F^2 = \max_U \|Y_{(1)} - UZ_{(1)}\|_F^2 = \max_U \mathrm{tr}\ Z_{(1)}Y_{(1)}^\top U$
    - 2乗して $\max_U \mathrm{tr}\ U^\top Y_{(1)} Z_{(1)}^\top Z_{(1)} Y_{(1)}^\top U$ s.t. $U^\top U = I$
      - これは固有値問題になる
  - $\mathcal{G}$ の最適解は $\mathcal{G} = \mathcal{X} \times_1 U^\top \times_2 V^\top \times_3 W^\top$
- U, V, W について一回づつ解いて、最後にを求めて終わるものが高階SVD（HOSVD）、収束するまで繰り返すものを高階直行反復（HOOI）と呼ぶ

凸最適化問題としてのテンソル分解：トレースノルムをテンソルに拡張する

- テンソル分解の最適化問題は凸では無い
- 行列の場合はトレースノルム（特異値の和）を用いることでランク制約を凸集合として入れることができた
- トレースノルムをテンソルに拡張する
  - 行列化したもののトレースノルムを入れる

$$\text{minimize} \ \|\mathcal{O}^*(\mathcal{X} - \mathcal{Y})\|_F^2 \ \ \text{s.t.} \ \textstyle\sum_n \|Y_{(n)}\|_* \le c$$

例えば：On the extension of trace norm to tensors. R.Tomioka, K.Hayashi, and H.Kashima, NIPS2010 Workshop: Tensors, kernels and machine learning, 2010.

テンソル分析の応用

ソフトウェア：Matlabでの実装が公開されている

- Matlabのツールボックスとして公開されている
  - Tensor Toolbox
  - *N*-way Toolbox

事例

- ソーシャルネットワーク分析
- Webリンク解析
- タグ推薦
- 脳波解析
- 画像認識