Московский физико-технический институт (НИУ)

Физтех-школа прикладной математики и информатики

Методы оптимизации

---

# Восстановление матриц

---

*Преподаватель:* Анна Руденко

*Студент:* Владислав Пыж

*Специальность:* Прикладная математика и информатика

*Семестр:* Осень 2021, 5 семестр

*E-Mail:* pyzh.va@phystech.edu

Долгопрудный, 2021

# Содержание

# Вступление

Это работа посвящена проблеме восстановления матриц, которая в англоязычной литературе именуется как Matrix Completion. Суть задачи заключается в следующем: представим, что мы храним какие-то данные в виде $n \times m$ матрицы $Z$, при этом какие-то поля данной матрицы для нас утеряны, либо просто недоступны. Мы хотим получить её приближение $\widehat{Z}$. Понятно, что в исходной формулировке в качестве приближения можно случайно заполнить $\widehat{Z}$, поэтому на $Z$ накладываются определенные ограничения и данная задача сводится к задачам оптимизации.

# Постановка проблемы

Представим, что у нас есть $m \times n$ матрица $Z$, в которой отсутсвуют какие-то поля и мы хотим приблизить матрицу $Z$ с помощью матрицы $\widehat{Z}$, накладывая какие-то ограничения. Более формально мы решаем проблему оптимизации, которая выглядит следующим образом:

$$\widehat{Z} = \arg \min_{M \in \mathbb{R}^{m \times n}} ||Z - M||_F^2 \text{ s.t. } \Phi(M) < C$$

Здесь $|| \cdot ||_F^2$ это квадрат нормы Фробениуса, которая равна сумме квадратов всех элементов матрицы (в numpy она вызывается изначально у linalg.norm), а $\Phi$ функция, которая должна отвечать за то, чтобы матрица $M$ в каком-то смысле была разреженной. Во многом то, что мы имеем в виду под этим понятием ведет к различным решениям поставленной задачи. Короткая сводка представлена в данной таблице.

| | Constraint | Resulting method |
|---|---|---|
| (a) | $\|\widehat{\mathbf{Z}}\|_{\ell_1} \leq c$ | Sparse matrix approximation |
| (b) | $\text{rank}(\widehat{\mathbf{Z}}) \leq k$ | Singular value decomposition |
| (c) | $\|\widehat{\mathbf{Z}}\|_\star \leq c$ | Convex matrix approximation |
| (d) | $\widehat{\mathbf{Z}} = \mathbf{UDV}^T$, $\Phi_1(\mathbf{u}_j) \leq c_1,\ \Phi_2(\mathbf{v}_k) \leq c_2$ | Penalized SVD |
| (e) | $\widehat{\mathbf{Z}} = \mathbf{LR}^T$, $\Phi_1(\mathbf{L}) \leq c_1,\ \Phi_2(\mathbf{R}) \leq c_2$ | Max-margin matrix factorization |
| (f) | $\widehat{\mathbf{Z}} = \mathbf{L} + \mathbf{S}$, $\Phi_1(\mathbf{L}) \leq c_1,\ \Phi_2(\mathbf{S}) \leq c_2$ | Additive matrix decomposition |

# Ядерная норма и решение задачи

В рамках данной работы в качестве $\Phi$ мы будем рассматривать только ограничение ранга, получившейся матрицы и ограничение на ядерную норму. Переформулируя это менее формально, теперь мы хотим получить приближение $\widehat{Z}$, при этом минимизируя ранк получившейся матрицы. В терминах этих ограничений наша задача принимает примерно следующий естесвенный вид:

$$\text{minimisize rank(M) subject to } m_{ij} = z_{ij} \text{ for all i, j}$$

Стоит однако заметить, что в таком виде мы получаем NP полную задачу. Из-за этого, а так же из соображений, что жесткое равенство может привести к перебучению, мы будем решать задачу с небольшой ошибкой:

$$\text{minimisize rank(M) subject to } \sum_{(i,j) \in \Omega} (z_{ij} - m_{ij})^2 \leq \delta$$

Это в свою очередь эквивалентно следующей задаче оптимизации:

$$\min_{rank(M)<r} \text{ subject to } \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2$$

К сожалению, это невыпуклая оптимизация, поэтому решения в общем виде мы получить не сможем. Однако стоит заметить, что если мы заменим ранг на ядерную норму, то задача станет выпуклой. Во многих ситуациях ядерная норма является эффективной релаксацией для ранга. Таким образом с помощью Лангранжиана мы свели нашу задачу к следующей:

$$\min_M \frac{1}{2} \sum_{(i,j)\in\Omega} (z_{ij} - m_{ij})^2 + \lambda \cdot ||M||_*$$

Теперь опишем алгоритм, который позволяет решить данную задачу.

## Описание алгоритма SVT

Введем некоторые обозначения. Пусть

$$S_\lambda(W) = UD_\lambda V, \text{ где } D_\lambda = diag[(d_i - \lambda)_+],$$
$$X_+ = \max(X, 0), UDV - \text{сингулярное разложение W}$$

Под $P_\Omega(Z)$ будем иметь в виду проекцию матрицы Z на множество $\Omega$. Теперь в этих терминах мы получаем следующий алгоритм, сходяшийся к минимуму:

**Algorithm 1** SOFT-IMPUTE

1. Initialize $Z^{\text{old}} = 0$.
2. Do for $\lambda_1 > \lambda_2 > \ldots > \lambda_K$:
   (a) Repeat:
      i. Compute $Z^{\text{new}} \leftarrow \mathbf{S}_{\lambda_k}(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$.
      ii. If $\frac{\|Z^{\text{new}} - Z^{\text{old}}\|_F^2}{\|Z^{\text{old}}\|_F^2} < \varepsilon$ exit.
      iii. Assign $Z^{\text{old}} \leftarrow Z^{\text{new}}$.
   (b) Assign $\hat{Z}_{\lambda_k} \leftarrow Z^{\text{new}}$.
3. Output the sequence of solutions $\hat{Z}_{\lambda_1}, \ldots, \hat{Z}_{\lambda_K}$.

Его реализацию и оценку работы можно видеть в прикрепленном блокноте. Сам алгоритм, я взял из работы [2].

# Другие алгоритмы решения данной задачи

Приведем так же очень похожий алгоритм, основывающейся на той же идее, но при этом, показывающий гораздо большую эффективность. Подробное описание данного алгоритма можно посмотреть в [3].

**Algorithm 1:** Singular Value Thresholding (SVT) Algorithm

**Input:** sampled set $\Omega$ and sampled entries $\mathcal{P}_\Omega(\boldsymbol{M})$, step size $\delta$, tolerance $\epsilon$, parameter $\tau$, increment $\ell$, and maximum iteration count $k_{\max}$
**Output:** $\boldsymbol{X}^{\text{opt}}$
**Description:** Recover a low-rank matrix $\boldsymbol{M}$ from a subset of sampled entries

Set $\boldsymbol{Y}^0 = k_0 \delta \mathcal{P}_\Omega(\boldsymbol{M})$ ($k_0$ is defined in (5.3))
Set $r_0 = 0$
**for** $k = 1$ **to** $k_{\max}$
    Set $s_k = r_{k-1} + 1$
    **repeat**
        Compute $[\boldsymbol{U}^{k-1}, \boldsymbol{\Sigma}^{k-1}, \boldsymbol{V}^{k-1}]_{s_k}$
        Set $s_k = s_k + \ell$
    **until** $\sigma^{k-1}_{s_k - \ell} \le \tau$
    Set $r_k = \max\{j : \sigma^{k-1}_j > \tau\}$
    Set $\boldsymbol{X}^k = \sum_{j=1}^{r_k} (\sigma^{k-1}_j - \tau) \boldsymbol{u}^{k-1}_j \boldsymbol{v}^{k-1}_j$

    **if** $\|\mathcal{P}_\Omega(\boldsymbol{X}^k - \boldsymbol{M})\|_F / \|\mathcal{P}_\Omega \boldsymbol{M}\|_F \le \epsilon$ **then break**

    Set $Y^k_{ij} = \begin{cases} 0 & \text{if } (i,j) \notin \Omega, \\ Y^{k-1}_{ij} + \delta(M_{ij} - X^k_{ij}) & \text{if } (i,j) \in \Omega \end{cases}$
**end** *for* $k$
Set $\boldsymbol{X}^{\text{opt}} = \boldsymbol{X}^k$

Так же сущетсвует несколько алгоритмов, решающих задачу поиска минимума суммы квадратов локально (т.к. вообще данная задача невыпуклая). Реализация одного из таких алгоритмов, основывающийся на [4] и [5], можно увидеть в прикрепленном блокноте.

# Вывод

В ходе данного проекта было показано как задача восстановления матрицы связана с задачами оптимизации. NP-полная задача поиска матрицы была сведена к выпуклой задаче, рассмотрено несколько алгоритмов её решения, сравнена их эффективность. Первый алгоритм показал себе хуже двух остальных, получая достаточно неточные предсказания отсутсвующих элементов. При решении задачи восстановления матрицы маленького ранга оба оставшихся алгоритма продемонстрировали себя неплохо. При этом в задачах разного типа, они показали разную эффективность. Было слаба изучена эффективность восстановления матриц в численном виде, таким образом вместе со специализацией алгоритмов это может быть перспективным развитием проекта.

# Источники

[1] Книга Trevor Hastie, Robert Tibshirani, Martin Wainwright: Statistical Learning with Sparsity ссылка

[2] Статья Rahul Mazumder, Trevor Hastie, Robert Tibshirani: Spectral Regularization Algorithms for Learning Large Incomplete Matrices ссылка

[3] Статья Jian-Feng Cai, Emmanuel J. Cand‘es, Zuowei Shen: A Singular Value Thresholding Algorithm for Matrix Completion ссылка

[4] Статья Ruslan Salakhutdinov, Andriy Mnih : Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo ссылка

[5] Статья Yehuda Koren, Robert Bell and Chris Volinsky: Matrix Factorization Techniques for Recommender Systems ссылка