Алгоритмы вычисления разложения Таккера и тензорного поезда

Команда “Matrix Masters”

Подготовили:
1) Середа Константин
2) Николаев Олег
3) Гавриш Борис
4) Матевосова Анастасия

Москва, 2023

Постановка задачи

Для хранения трехмерного тензора требуется N^3 ячеек памяти, что довольно много. Однако иногда в практических задачах данные обладают структурой, позволяющей записать их малопараметрическое представление, требующее меньше ячеек в памяти компьютера. Как известно, базовыми тензорными разложениями являются: каноническое, разложение Таккера и тензорный поезд.

Но как “наиболее оптимально” вычислять данные разложения?

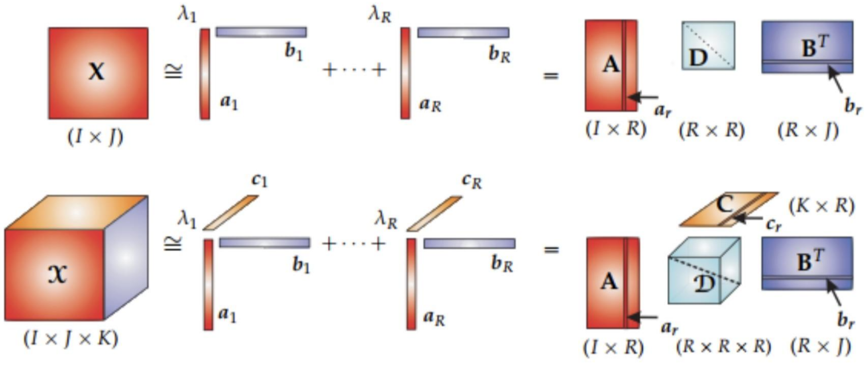
Задача: изучить различные вариации разложения Таккера и тензорного поезда на тензорах специального вида.
Гипотеза: алгоритмы с модифицированным подсчетом SVD-разложения работают быстрее и дают примерно то же качество, что и стандартные библиотечные разложения.
Оценка качества: измерение времени работы и Фробениусовой нормы приращения.

Каноническое разложение (CP)

$$A(i,j,k) = \sum_{\alpha=1}^{R} u_\alpha(i) v_\alpha(j) w_\alpha(k) = \sum_{\alpha=1}^{R} u(\alpha,i) v(\alpha,j) w(\alpha,k)$$

- Хорошее сжатие данных: из N^3 в 3NR;
- Основной недостаток CP-формата: трудность вычисления точного значения канонического ранга R, которое часто сводится к решении NP-трудной задачи;
- Поэтому на практике обычно получают приближенное каноническое разложение с помощью **итерационного метода переменных наименьших квадратов**, фиксируя целевое значение ранга R.

Про алгоритм переменных наименьших квадратов нет доказанных теорем, гарантирующих его сходимость в общем случае, поэтому могут возникать проблемы со скоростью сходимости и обоснованием результатов.



Algorithm Схема процедуры переменных наименьших квадратов(A, R)

$U(\alpha,i), V(\alpha,j), W(\alpha,k) \leftarrow$ инициализация случайными значениями
repeat
1. Зафиксируем значения $V(\alpha,j), W(\alpha,k)$,
$U(\alpha,i) \leftarrow$ решим задачу линейных наименьших квадратов $\|\sum_{\alpha=1}^{R} u(\alpha,i)v(\alpha,j)w(\alpha,k) - A\|_F$.
2. Зафиксируем значения $U(\alpha,i), W(\alpha,k)$,
$V(\alpha,j) \leftarrow$ решим задачу линейных наименьших квадратов $\|\sum_{\alpha=1}^{R} u(\alpha,i)v(\alpha,j)w(\alpha,k) - A\|_F$.
3. Зафиксируем значения $U(\alpha,i), V(\alpha,j)$,
$W(\alpha,k) \leftarrow$ решим задачу линейных наименьших квадратов $\|\sum_{\alpha=1}^{R} u(\alpha,i)v(\alpha,j)w(\alpha,k) - A\|_F$.

until stoppingcriteria

Разложение Таккера (HOSVD)

$$T(i,j,k) = \sum_{\alpha_1=1}^{r_1}\sum_{\alpha_2=1}^{r_2}\sum_{\alpha_3=1}^{r_3} G(\alpha_1,\alpha_2,\alpha_3)U(\alpha_1,i)V(\alpha_2,j)W(\alpha_3,k)$$

r_1, r_2, r_3 — ранги Таккера;

$G(\alpha_1,\alpha_2,\alpha_3)$ — ядро Таккера;

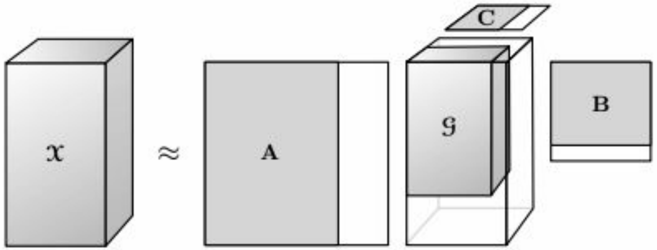
$U(\alpha_1,i), V(\alpha_2,j), W(\alpha_3,k)$ — факторы разложения.

Пусть дан тензор $A \in \mathbb{C}^{N_1 x N_2 x N_3}$

Алгоритм HOSVD:

- Определим матрицы разверток $A_{i,jk} \in \mathbb{C}^{N_1 x N_2 N_3}, A_{j,ik} \in \mathbb{C}^{N_2 x N_1 N_3}, A_{k,ij} \in \mathbb{C}^{N_3 x N_1 N_2}$
- $A_{i,jk} = U_1\Sigma_1 V_1^*,\ A_{j,ik} = U_2\Sigma_2 V_2^*,\ A_{k,ij} = U_3\Sigma_3 V_3^*$
- $G = U_1^* U_2^* U_3^* A$

$i \in 0, \dots, N_1 - 1; j \in 0, \dots, N_2 - 1; k \in 0, \dots, N_3 - 1.$



Последовательное многомерное SVD (st-HOSVD)

Пусть дан тензор $A \in \mathbb{C}^{N_1 x N_2 x N_3}$

Алгоритм st-HOSVD:

- reshape: $A \to A_{i,jk}$
- $A_{i,jk} = U_1 \Sigma_1 V_1^*$; $U = U_1$, $A_{\alpha_1,jk} = \Sigma_1 V_1^*$
- reshape: $A_{\alpha_1,jk} \to A_{\alpha_1 j,k}$
- $A_{\alpha_1 j,k} = U_2 \Sigma_2 V_2^*$; $W = V_2$, $A_{\alpha_1 j,\alpha_3} = U_2 \Sigma_2$
- reshape: $A_{\alpha_1 j,\alpha_3} \to A_{j,\alpha_1\alpha_3}$
- $A_{j,\alpha_1\alpha_3} = U_3 \Sigma_3 V_3^*$; $V = U_3$, $A_{\alpha_2,\alpha_1\alpha_3} = \Sigma_3 V_3$
- reshape: $A_{\alpha_2,\alpha_1\alpha_3} \to G_{\alpha_1,\alpha_2,\alpha_3}$

$\longrightarrow$

Алгоритм randomized SVD:

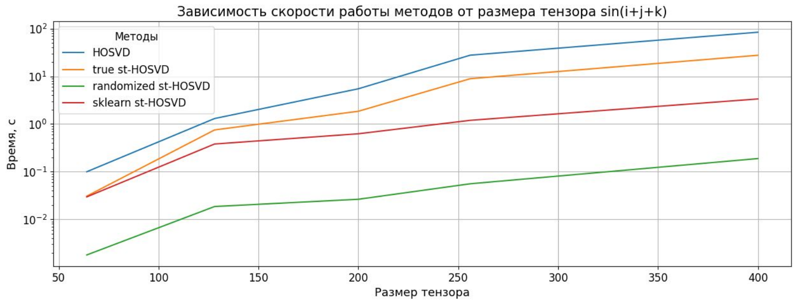
- Сгенерировать случайную матрицу $\Omega \in \mathbb{C}^{N x r}$, где $\Omega_{i,j} \sim N(0,1)$
- Вычислить $B = A\Omega \to O(MNr)$
- Сделать QR-разложение: $B = QR \to O(Mr^2)$
- Вычислить SVD от $Q^* A = U\Sigma V^* \to O(MNr + Nr^2)$
- $\tilde{A}_r = Q(Q^* A) = (QU)\Sigma V^* = \tilde{U}\Sigma V^* \to O(MNr)$

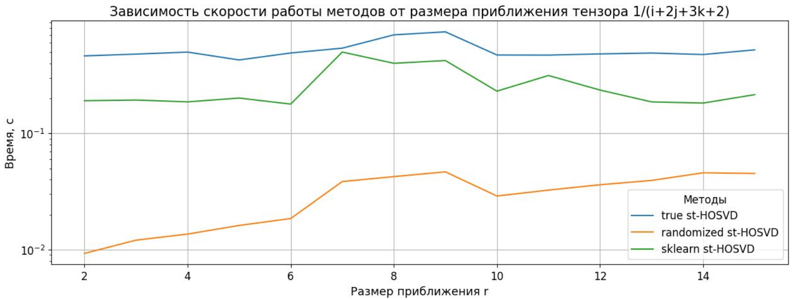
Пусть дан тензор $A \in \mathbb{C}^{N_1 x N_2 x N_3}$

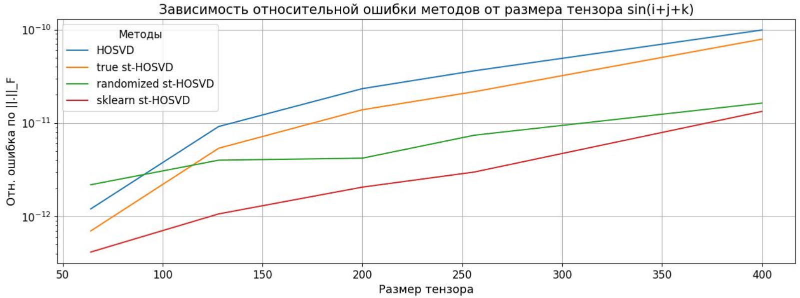
Алгоритм randomized st-HOSVD:

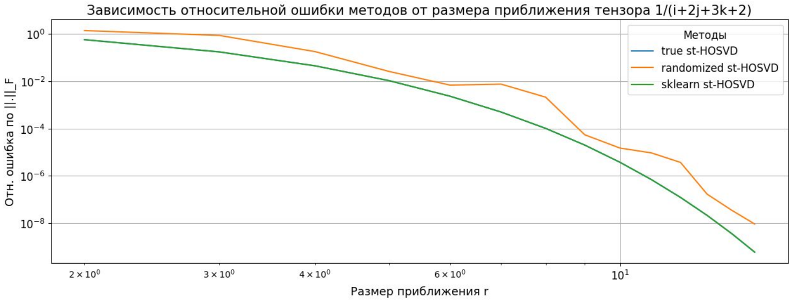
- reshape: $A \to A_{i,jk}$
- $U_1, \Sigma_1, V_1 = randSVD(A_{i,jk})$; $U = U_1$, $A_{\alpha_1,jk} = \Sigma_1 V_1^*$
- reshape: $A_{\alpha_1,jk} \to A_{\alpha_1 j,k}$
- $U_2, \Sigma_2, V_2 = randSVD(A_{\alpha_1 j,k})$; $W = V_2$, $A_{\alpha_1 j,\alpha_3} = U_2 \Sigma_2$
- reshape: $A_{\alpha_1 j,\alpha_3} \to A_{j,\alpha_1\alpha_3}$
- $U_3, \Sigma_3, V_3 = randSVD(A_{j,\alpha_1\alpha_3})$; $V = U_3$, $A_{\alpha_2,\alpha_1\alpha_3} = \Sigma_3 V_3$
- reshape: $A_{\alpha_2,\alpha_1\alpha_3} \to G_{\alpha_1,\alpha_2,\alpha_3}$

Эксперименты с разложениями в формате Таккера







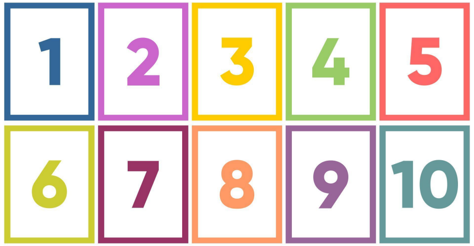


Вывод: во всех экспериментах по времени лидирует st-HOSVD с randSVD, а по ошибке в первом случае соответствующий алгоритм из sklearn, а во втором - он же вместе с true st-HOSVD.

Приложения разложения Таккера

Сжатие изображений

Исходное изображение



Rank_tucker=25

Rank_tucker=15

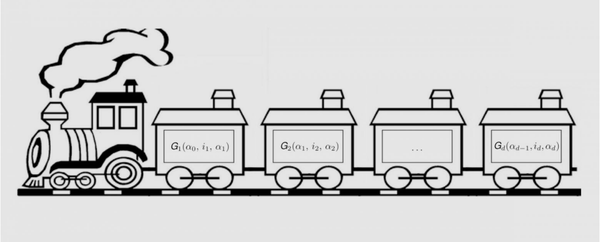
Rank_tucker=10

Rank_tucker=5

а также
-Рекомендательные сервисы
-Графы с временной структурой
-Анализ биомедицинских данных
и др.

Тензорный поезд (TT)

$$K(i_1,\dots,i_d) \approx \sum_{\alpha_0=1}^{1} \sum_{\alpha_1=1}^{r_1} \dots \sum_{\alpha_d=1}^{1} G_1(\alpha_0,i_1,\alpha_1) G_2(\alpha_1,i_2,\alpha_2) \dots \\ \dots G_d(\alpha_{d-1},i_d,\alpha_d)$$



Ключевые особенности:

a. не игнорирует аналитическую связь между компонентами тензора и их индексами
b. в среднем, хорошая сходимость, устойчивость
c. *O(rnd)* по памяти (где r — это средний ранг ядер, d — размерности пространства)
d. в большинстве случаев дает наименьший возможный TT-rank
e. развитая тензорная алгебра (умножение, сложение)

Сложности:

a. представление не единственно
b. алгоритм может привести к разным рангам результирующего тензора
c. быстрый рост рангов ядер в случае, если размер множества образов, которым обладают производящие функции, слишком велик

Область применения:

Решение многомерных параболических уравнений в частных производных

Лучше всего работает (по сравнению с другими методами):

a. разреженные матрицы
b. матрицы, которые имеют небольшое количество возможных значений

Require: d-dimensional tensor $\mathbf{A}$, prescribed accuracy ε.
Ensure: Cores $G_1,\dots,G_d$ of the TT-approximation $\mathbf{B}$ to $\mathbf{A}$ in the TT-format with TT-ranks $\widehat{r}_k$ equal to the δ-ranks of the unfoldings A_k of $\mathbf{A}$, where $\delta = \frac{\varepsilon}{\sqrt{d-1}}||A||_F$. The computed approximation satisfies

$$||\mathbf{A}-\mathbf{B}||_F \le \varepsilon||\mathbf{A}||_F.$$

{Initialization}
Compute truncation parameter $\delta = \frac{\varepsilon}{\sqrt{d-1}}||\mathbf{A}||_F$.
Temporary tensor: $\mathbf{C} = \mathbf{A}$, $r_0 = 1$.
for $k = 1$ to $d-1$ **do**
  $C := \mathrm{reshape}(C, [r_{k-1}n_k, \frac{\mathtt{numel}(C)}{r_{k-1}n_k}])$.
  Compute δ-truncated SVD: $C = USV + E, ||E||_F \le \delta, r_k = \mathrm{rank}_\delta(C)$.
  New core: $G_k := \mathrm{reshape}(U, [r_{k-1}, n_k, r_k])$.
  $C := SV^\top$.
end for
$G_d = C$.
Return tensor $\mathbf{B}$ in TT-format with cores $G_1,\dots,G_d$.

**Tensor-Train Decomposition, 2011, I. V. Oseledets*

Тензорный поезд: алгоритм разложения

Algorithm 1 TT-SVD

Input: $X \in \mathbf{R}^{n_1 \times \cdots \times n_d}$, max. TT-rank $r_{\max} \geq 1$, tolerance ϵ

Output: TT decomposition $\sum_{j_1,\ldots,j_{d-1}} T^{(1)}_{1,i_1,j_1} T^{(2)}_{j_1,i_2,j_2} \cdots T^{(d)}_{j_{d-1},i_d,1} = \tilde{X}_{i_1,\ldots,i_d}$ with $\|X - \tilde{X}\|_F \leq \epsilon \|X\|_F$ if $r_{\max} \geq r_\delta^{(i)}$

1: $\delta \leftarrow \frac{\epsilon}{\sqrt{d-1}} \|X\|_F$ *(truncation parameter)*
2: $W \leftarrow X$ *(temporary tensor)*
3: $\bar{n} \leftarrow \prod_{i=1}^{d} n_i$ *(total size of W)*
4: $r_d \leftarrow 1$
5: **for** $i = d, \ldots, 2$ **do**
6: $W \leftarrow \text{reshape}\left(W, \left(\frac{\bar{n}}{n_i r_i} \quad n_i r_i\right)\right)$
7: Calculate SVD: $U \Sigma V^T = W$ with $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_{n_i r_i})$
8: Choose rank $r_{i-1} = \min(r_{\max}, r_\delta^{(i)})$, $r_\delta^{(i)} = \min\left(j : \sigma_{j+1}^2 + \sigma_{j+2}^2 + \cdots \leq \delta^2\right)$
9: $T^{(i)} \leftarrow \text{reshape}\left((V_{:,1:r_{i-1}})^T, \left(r_{i-1} \quad n_i \quad r_i\right)\right)$
10: $\bar{n} \leftarrow \frac{\bar{n} r_{i-1}}{n_i r_i}$ *(new total size of W)*
11: $W \leftarrow U_{:,1:r_{i-1}} \text{diag}(\sigma_1, \ldots, \sigma_{r_{i-1}})$
12: **end for**
13: $T^{(1)} \leftarrow \text{reshape}\left(W, \left(1 \quad n_1 \quad r_1\right)\right)$

Модификация: Optimized TSQR TT-SVD

Calculate R from the QR decomposition: $QR = W^{(i)}$
Calculate small SVD: $\bar{U}\Sigma V^T = R$ with $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_{n_i r_i})$

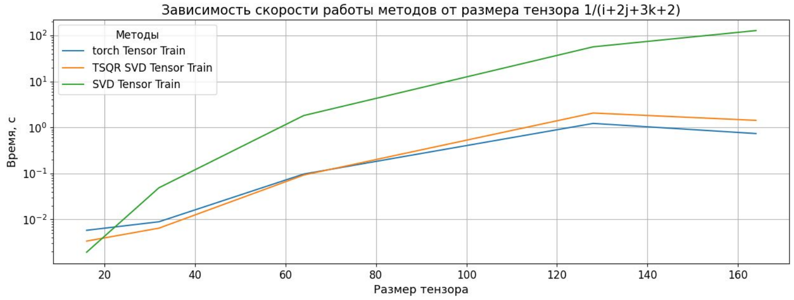
$$W^{(i-1)} \leftarrow \text{reshape}\left(W^{(i)} V_{:,1:r_{i-1}}, \left(\frac{\bar{n}}{n_{i-1} r_{i-1}} \quad n_{i-1} r_{i-1}\right)\right)$$

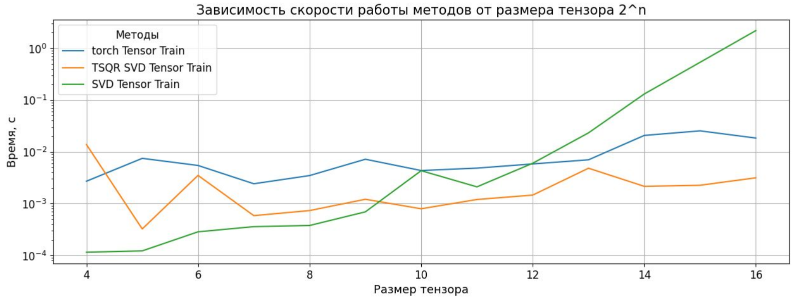
Доказано, что данный алгоритм позволяет получить квази-оптимальное разложение

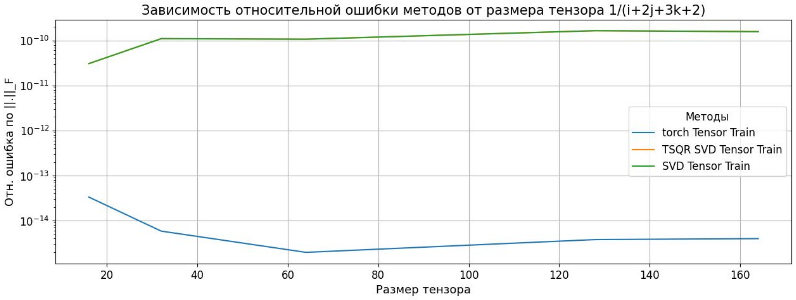
Corollary 2.4. *Given a tensor* $\mathbf{A}$ *and rank bounds* r_k, *the best approximation to* $\mathbf{A}$ *in the Frobenius norm with TT-ranks bounded by* r_k *always exists (denote it by* $\mathbf{A}^{best}$*), and the TT-approximation* $\mathbf{B}$ *computed by the TT-SVD algorithm is quasi-optimal:*

$$||\mathbf{A} - \mathbf{B}||_F \leq \sqrt{d-1} ||\mathbf{A} - \mathbf{A}^{best}||_F.$$

Эксперименты с реализациями TT







Вывод: модификация алгоритма дает значительное увеличение скорости вычислений; точность реализованных методов ниже, чем у библиотечного; наблюдается различие в результатах в зависимости от типа матрицы.

Приложение TT в кооперативной Теории игр

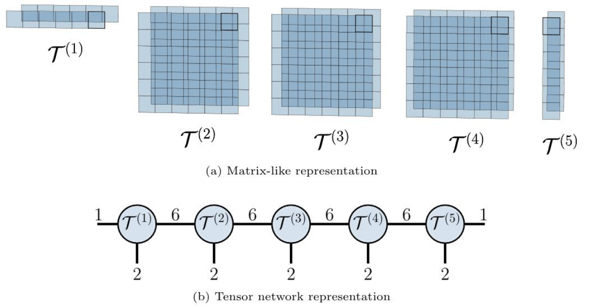
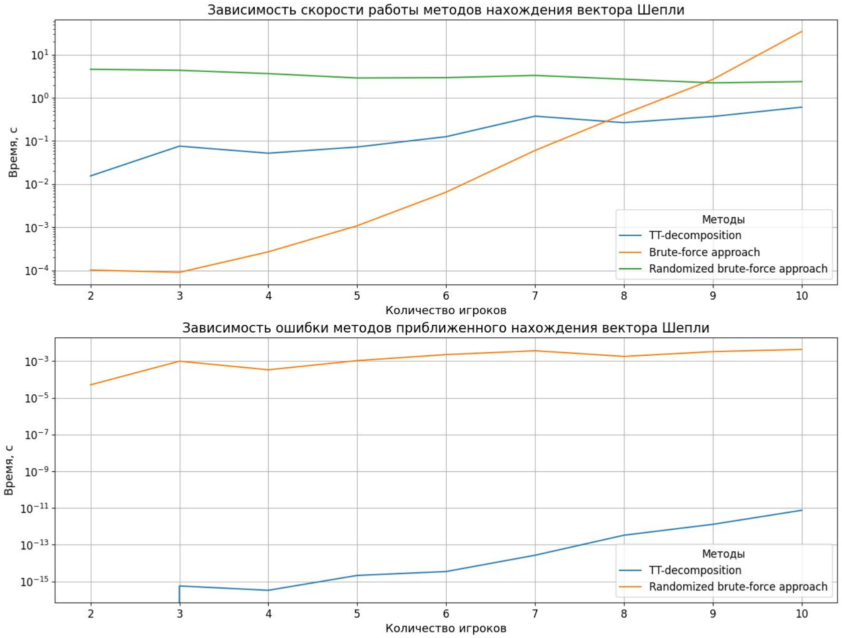


Fig. 1. A TT cooperative game with $|N| = 5$ players encodes the utility function v in five *TT cores*, each of which consists of two matrices. The value for every possible coalition $S \subseteq N$ is represented by the $2^{|N|} = 32$ possible matrix product sequences. In this example, the TT ranks are all equal to 6. In (b) we show each core as a node in a graph (a so-called tensor network).



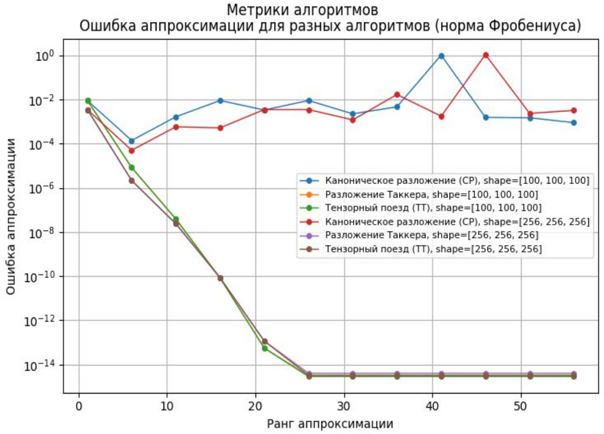
<u>Специфика задачи:</u>

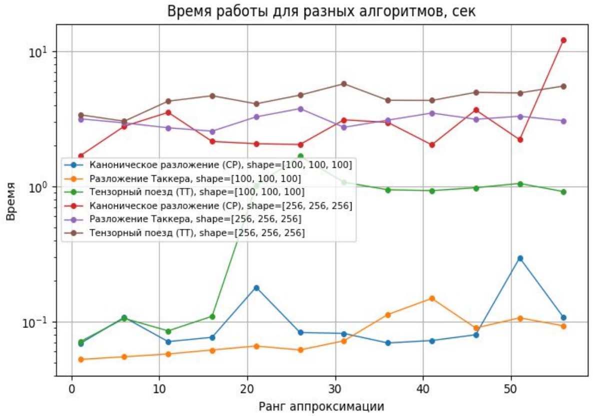
Тензор особого вида: по каждому измерению только два индекса. Имеет важные практические приложения, так как кооперативные игры могут использоваться для распределения платы при совместном использовании ресурсов.

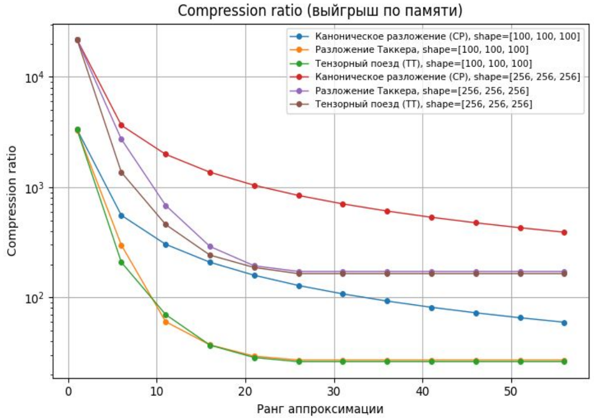
Вывод: метод позволяет с высокой точностью аппроксимировать истинное значение и при этом выполнять вычисления для большего числа игроков.

Сравнение скоростей работы, потребления памяти, точности аппроксимации в зависимости от ранга

Аналитическая связь №1: $\sqrt{\sqrt{X} \cdot (Y+Z) + Y \cdot Z^2} \cdot (X + \sin(Y) \cdot \cos(Z))$

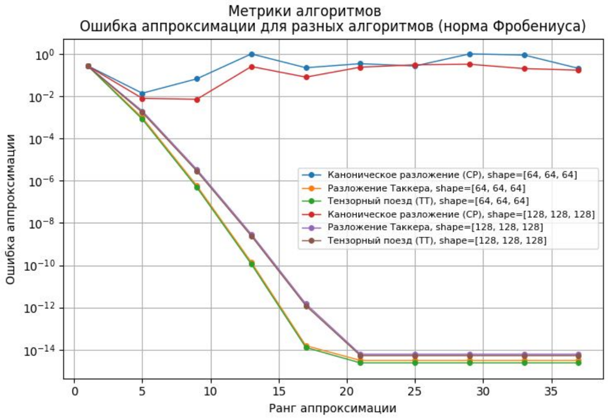


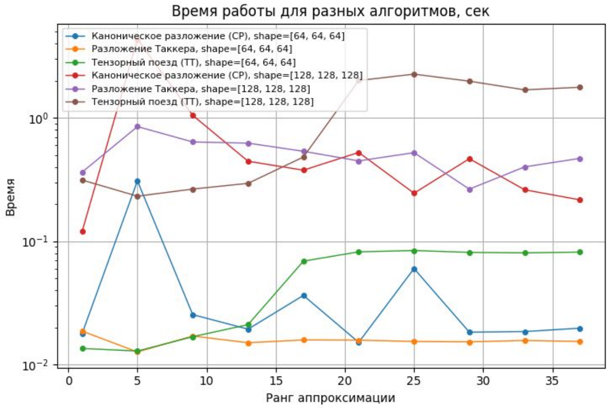


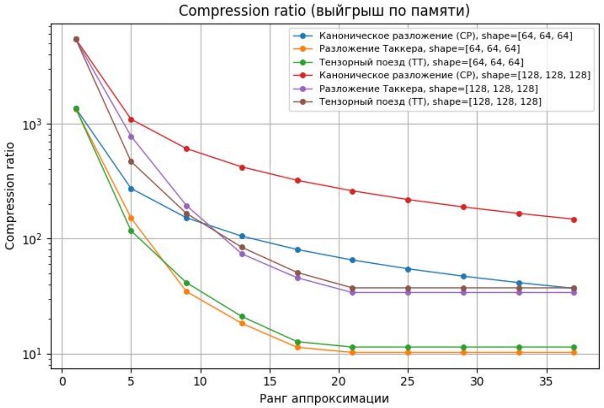


Сравнение скоростей работы, потребления памяти, точности аппроксимации в зависимости от ранга

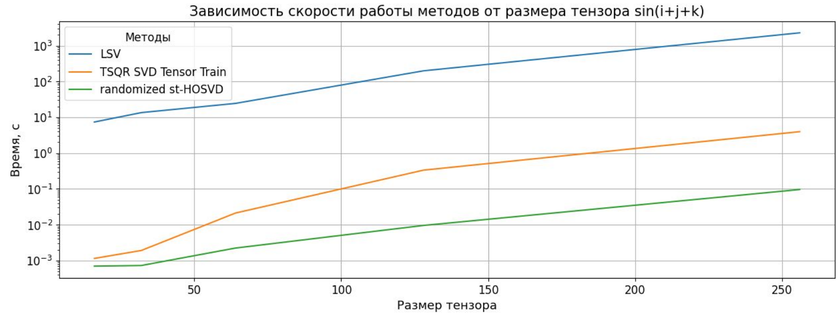
Аналитическая связь №2: $\frac{1}{X+Y+Z+1}$

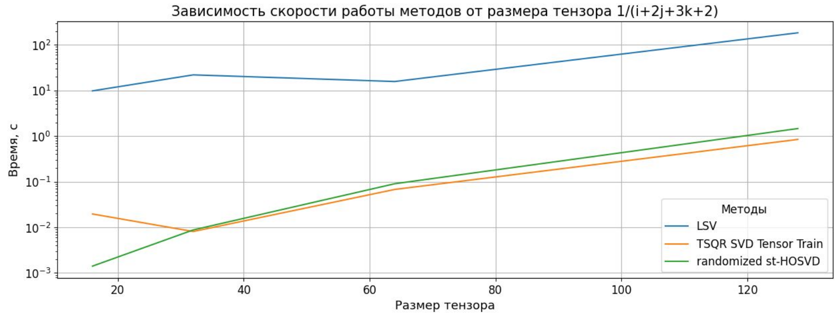


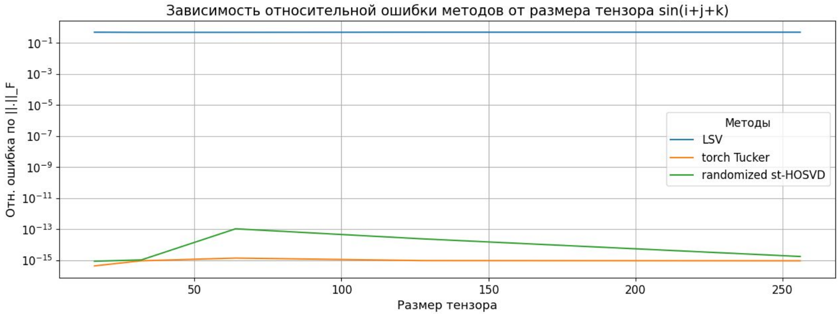


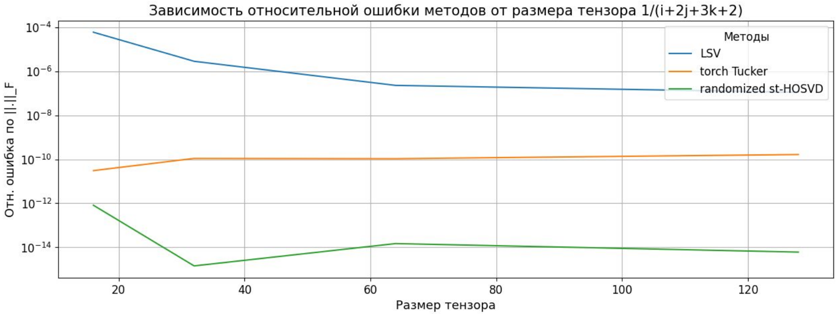


Финальный эксперимент









Вывод: приближенное CP имеет сложности со сходимостью на тензорах sin(i+j+k), а на 1/(i+2j+3k+2) наблюдается заметно лучшая аппроксимация. Два других алгоритма везде демонстрируют примерно одинаковую ошибку, причем скорость их работы зависит от ранга тензора: для малоранговых быстрее st-HOSVD с randSVD, для тензоров полного ранга - TSQR SVD TT.

Итоги

Результаты: в ходе работы мы разобрались с алгоритмами вычисления разложения Таккера и тензорного поезда (TT), запрограммировали их на Python. Затем провели эксперименты по сравнению различных реализаций как между собой, так и друг с другом на тензорах специального вида и тензорах из приложений.

Выводы: 1. Методы малоранговой аппроксимации Таккера и TT демонстрируют хорошее качество и время работы на различных тензорах, а также обеспечивают существенный выигрыш в памяти.
2. Лучше всего себя показали самописные алгоритмы randomized st-HOSVD и TSQR SVD TT на тензорах больших и малых рангов.
3. Оба разложения активно применяются в приложениях: Таккер в графах с временной структурой, рекомендательных сервисах, TT в теории игр, решении многомерных параболических уравнений в частных производных.

План на будущее: сравнить все полученные алгоритмы на тензорах с другой спецификой.

Рассматриваемая литература и ссылка на GitHub

- Oseledets I. V., Tyrtyshnikov E. E. (2009). Breaking the curse of dimensionality, or how to use SVD in many dimensions. SIAM Journal on Scientific Computing, 31(5), 3744-3759.
- Оселедец, И. В. Вычислительные тензорные методы и их применения. докторская диссертация, М.: ИВМ РАН, 2012.
- Oseledets, Ivan. (2011). Tensor-Train Decomposition. SIAM J. Scientific Computing. 33. 2295-2317. 10.1137/090752286.
- Rafael Ballester-Ripoll. Tensor approximation of cooperative games and their semivalues. International Journal of Approximate Reasoning. 142. 2022. 94-108
- Gleb Ryzhakov, Ivan Oseledets. Constructive TT-representation of the tensors given as index interaction functions with applications. arXiv. 2022
- Constructive TT-representation of the tensors given as index interaction functions with applications

Репозиторий GitHub с кодом проекта: https://github.com/ol3gka/AI_Masters_NLA_projects_Matrix_Masters

Спасибо за внимание!