# Подготовительный этап: на пути к отношению Реллея

## Основные определения

**Определение 1.** *Тензорным (полилинейным) отображением* назовём отображение $\mathcal{A}\colon \mathbb{R}^{n_1\times n_2\times\ldots\times n_d} \to \mathbb{R}^{m_1\times m_2\times\ldots\times m_d}$, линейное по каждому аргументу, при фиксированных прочих.

Такое отображение (при фиксированных базисах) можно задать тензором $\mathbf{A} \in \mathbb{R}^{(m_1\times n_1)\times\ldots\times(m_d\times n_d)}$, причём $\forall\, V \in \mathbb{R}^{m_1\times m_2\times\ldots\times m_d}\ \exists\, T \in \mathbb{R}^{n_1\times n_2\times\ldots\times n_d}$ такой что

$$V = \mathcal{A}(T) = \mathbf{A}\cdot T$$

где умножение тензорного оператора $\mathbf{A}$ на тензор $T$ осуществляется по правилу

$$(\mathbf{A}\cdot T)_{i_1,\ldots,i_d} = \sum_{j_1=1}^{n_1}\cdots\sum_{j_d=1}^{n_d}\mathbf{A}_{i1,j1,\ldots,i_d,j_d}\cdot T_{j_1,\ldots,j_d} \tag{1}$$

**Замечание.** Если source и target пространства совпадают, то будем называть $\mathcal{A}$ тензорным (полилинейным) оператором.

**Определение 2.** *Собственными значениями* тензорного оператора $\mathcal{A}$ называются значения $\lambda \in \mathbb{R}$, удовлетворяющие уравнению

$$\mathcal{A}(T) = \lambda T$$

Соответствующие $T \in \mathbb{R}^{n_1\times n_2\times\ldots\times n_d}\setminus\{0\}$ называются *собственными тензорами.*

**Определение 3.** Тензорный оператор $\mathcal{A}$, задаваемый тензором $\mathbf{A}$ называется *симметричным*, если $\forall\,(i_1,\ldots,i_d) \subset \mathbb{N}^{n_1\times\ldots\times n_d}, (j_1,\ldots,j_d) \subset \mathbb{N}^{m_1\times\ldots\times m_d}$ выполнено

$$\mathbf{A}_{i_1,j_1,\ldots,i_d,j_d} = \mathbf{A}_{j_1,i_1,\ldots,j_d,i_d}$$

**Определение 4.** *Скалярным произведением тензоров* называется симметричная, полилинейная форма $(\cdot,\cdot)\colon \mathbb{R}^{n_1\times n_2\times\ldots\times n_d}\times\mathbb{R}^{n_1\times n_2\times\ldots\times n_d}\to\mathbb{R}$, по правилу

$$(X, Y) \mapsto \sum_{i_1,\ldots,i_d} X_{i_1,\ldots,i_d} Y_{i_1,\ldots,i_d}$$

Здесь положительная определённость понимается в обычном смысле: $X \in \mathbb{R}^{n_1\times n_2\times\ldots\times n_d}\setminus\{0\} \implies (X, X) > 0$.

## Постановка задачи

Аналогично с матричным случаем, можно определить отношение Реллея следующим образом: если симметричный тензорный оператор $\mathcal{A}\colon \mathbb{R}^{n_1\times n_2\times\ldots\times n_d}\to\mathbb{R}^{n_1\times n_2\times\ldots\times n_d}$ задаётся тензором $\mathbf{A}$, то отношение Реллея:

$$R(T) = \frac{(T, \mathbf{A}T)}{(T, T)}$$

Так же аналогично с матричным случаем выполнено

$$\min \lambda_i = \min_{T \neq 0} R(T) \qquad\qquad \max \lambda_i = \max_{T \neq 0} R(T)$$

Эти значения могут быть найдены методами Римановой оптимизации (даже понятно, что конкретно надо делать). В чём же проблема? А проблем на самом деле несколько:

1. В пакете TTAX не реализовано умножение тензора на тензор;
2. Не ясно, откуда брать симметричные тензоры (кроме супердиагональных);

На практике вторая проблема наверняка и не проблема вовсе: я просто ещё не искал, как делать симметричные тензоры (понятно, что можно просто сделать $\mathbf{A}^T\mathbf{A}$, это не интересно). Первая проблема куда более серьёзная, ей и будем заниматься.

## Тензорный матвек

Чего мы в сущности ждём от операции, которую хотим реализовать? Запишем все наши желания для функции, реализующей операцию:

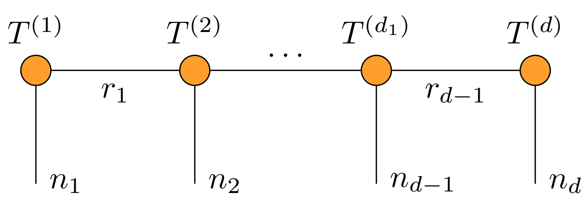
- Функция принимает два тензора в формате TT, один из которых — оператор;
- Функция нигде не должна собирать тензор целиком;
- Функция выполняет умножение оператора на тензор и возвращает ответ в TT формате;
- В функции не должно быть никаких side-эффектов, чтобы не мешать автодифференцированию;

**Определение 5.** Говорят, что тензор $T \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}$ имеет *TT-разложение* $\left[T^{(1)}, \ldots, T^{(d)}\right]$, если каждый его элемент может быть представлен в виде

$$T_{i_1,\ldots,i_d} = \sum_{k_0=1}^{r_0} \cdots \sum_{k_d=1}^{r_d} T^{(1)}_{k_0,i_1,k_1} \cdot \ldots \cdot T^{(d)}_{k_{d-1},i_d,k_d}$$

Тензоры $T^{(i)} \in \mathbb{R}^{r_{i-1} \times n_i \times r_i}$ называются *TT-ядрами*, а вектор $(r_1, \ldots, r_{d-1})$ называется *TT-рангом.*

Кроме того мы будем представлять TT-разложение тензора $T$ в виде тензорной сети следующим образом:



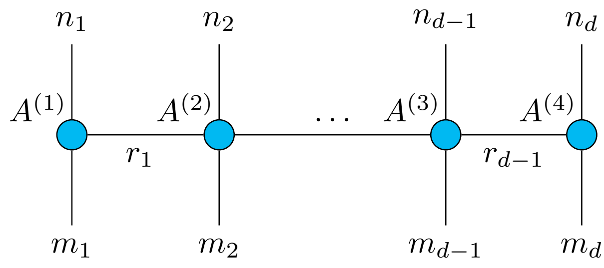

**Определение 6.** Будем говорить, что тензорный оператор $\mathbf{A} \in \mathbb{R}^{(m_1 \times n_1) \times \ldots \times (m_d \times n_d)}$ имеет *TT-разложение* $\left[A^{(1)}, \ldots, A^{(d)}\right]$, если каждый его элемент может быть представлен в виде

$$\mathbf{A}_{i_1,j_1,\ldots,i_d,j_d} = \sum_{k_0=1}^{r_0} \cdots \sum_{k_d=1}^{r_d} A^{(1)}_{k_0,i_1,j_1,k_1} \cdot \ldots \cdot A^{(d)}_{k_{d-1},i_d,j_d,k_d}$$

Соответственно тензоры $A^{(i)} \in \mathbb{R}^{r_{i-1}\times m_i \times n_i \times r_i}$ — его *TT-ядра*, вектор $(r_1, \ldots, r_{d-1})$ — его *TT-ранг*.

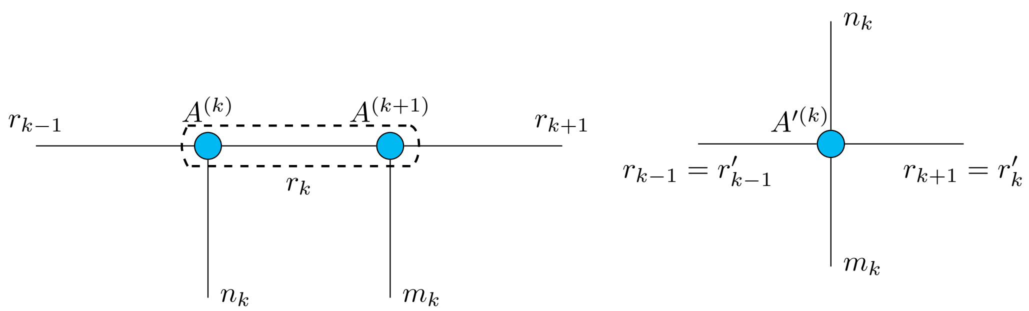
**Определение 7.** По соглашению значения $r_0$ и $r_d$ полагаются равными 1.

Будем представлять TT-разложение оператора $\mathbf{A}$ в виде тензорной сети следующим образом:




Сразу можно заметить ещё одну проблему: мы хотим, чтобы TT-ядра оператора являлись тензорами четвёртого порядка, в то время как TTAX позволяет нам хранить в ядрах только тензоры порядка 3. Чтобы решить эту проблему, опишем процедуру получения TT-разложения оператора с ядрами четвёртого порядка, имея на руках TT-разложение оператора с ядрами третьего порядка:

Сейчас тензорный оператор $\mathbf{A} \in \mathbb{R}^{m_1\times n_1 \times \ldots \times m_d \times n_d}$ представляется в TTAX в виде $\left[A^{(1)}, \ldots, A^{(2d)}\right]$, где $A^{(k)} \in \mathbb{R}^{r_{k-1}\times m_k \times r_k}$, а $A^{(k+1)} \in \mathbb{R}^{r_k \times n_k \times r_{k+1}}$. Имея два таких тензора порядка 3 мы можем свернуть их по общей моде и получить тензор порядка 4. Нагляднее покажем процесс на тензорной сети:




(a) Два ядра порядка 3 до свёртки (b) Полученное ядро порядка 4

Таким образом мы описали процедуру, как имея TT-разложение оператора в формате, предоставляемом TTAX получить TT-разложение в нужном нам формате.

Теперь рассмотрим операцию произведения тензорного оператора $\mathbf{A} \in \mathbb{R}^{(m_1\times n_1)\times \ldots \times (m_d \times n_d)}$ на тензор $T \in \mathbb{R}^{n_1\times \ldots \times n_d}$. Если расписать формулу (1), подставив на места тензоров $\mathbf{A}$ и $T$ соответствующие им TT-разложения, то станет ясно, что один и тот же индекс, по которому ведётся суммирование (скажем, $j_k$), встречается один раз в $A^{(k)}$, и один раз в $T^{(k)}$, и больше не встречается. Это значит, что произведение оператора на тензор может быть

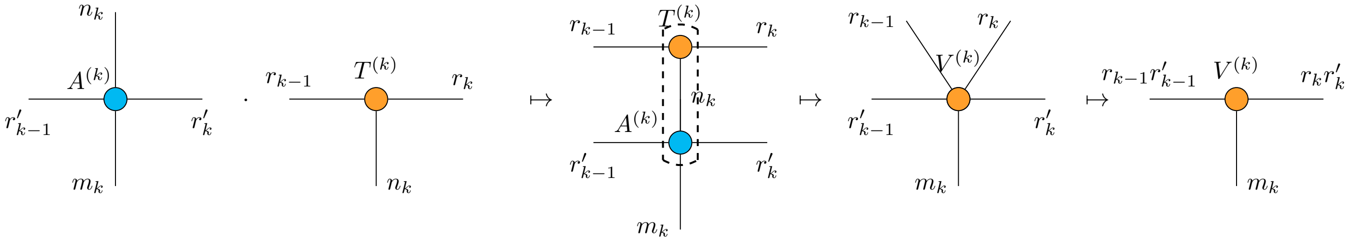



Рис. 2: Процесс умножения оператора на тензор $V = \mathbf{A} \cdot T$

записано в терминах произведения конкретных ТТ-ядер. Покажем процесс произведения оператора на тензор наглядно через тензорные сети:

Осталось это только заимплементить, и можно приступать к Римановой оптимизации.