Software and Tools for Computational Engineering

RWTH AACHEN UNIVERSITY

Einführung in die Programmierung mit C++

Übungsblatt 7

Tensorarithmetik und Iteratoren

Sebastian Christodoulou, Alexander Fleming und Uwe Naumann

Informatik 12:
Software and Tools for Computational Engineering (STCE)

RWTH Aachen

In diesem Übungsblatt arbeiten wir mit 3-dimensionalen Tensoren, d.h. mit Strukturen, die eine Dimension mehr haben, als eine Matrix. Gegeben sei ein Tensor

$$T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$$

Für die Verallgemeinerung der Multiplikation auf solchen Strukturen, definieren wir *Tensorkontraktion*. Gegeben seien Vektoren $u \in \mathbb{R}^{d_1}$, $v \in \mathbb{R}^{d_2}$ und $w \in \mathbb{R}^{d_3}$
Folgende Operationen sind dann definiert:

$$T \underset{\{1\}\{1\}}{\odot} u = A \quad \text{wobei } A \in \mathbb{R}^{d_2 \times d_3} \quad \text{und} \quad A(i,j) = \sum_{k=0}^{d_1} u(k) \cdot T(k,i,j)$$

$$T \underset{\{2\}\{1\}}{\odot} v = B \quad \text{wobei } B \in \mathbb{R}^{d_1 \times d_3} \quad \text{und} \quad B(i,j) = \sum_{k=0}^{d_2} v(k) \cdot T(i,k,j) \tag{1}$$

$$T \underset{\{3\}\{1\}}{\odot} w = C \quad \text{wobei } C \in \mathbb{R}^{d_1 \times d_2} \quad \text{und} \quad C(i,j) = \sum_{k=0}^{d_3} w(k) \cdot T(i,j,k)$$

wobei der operator $\underset{\{p\}\{q\}}{\odot}$ die p-te Dimension vom linken Input, und die q-te Dimension vom rechten Input *kontrahiert* ("zusammengezogen"). Die kohtrahierten Dimensionen müssen stets gleicher Länge sein.

Visualisuerung: Kontraktion einer Dimension

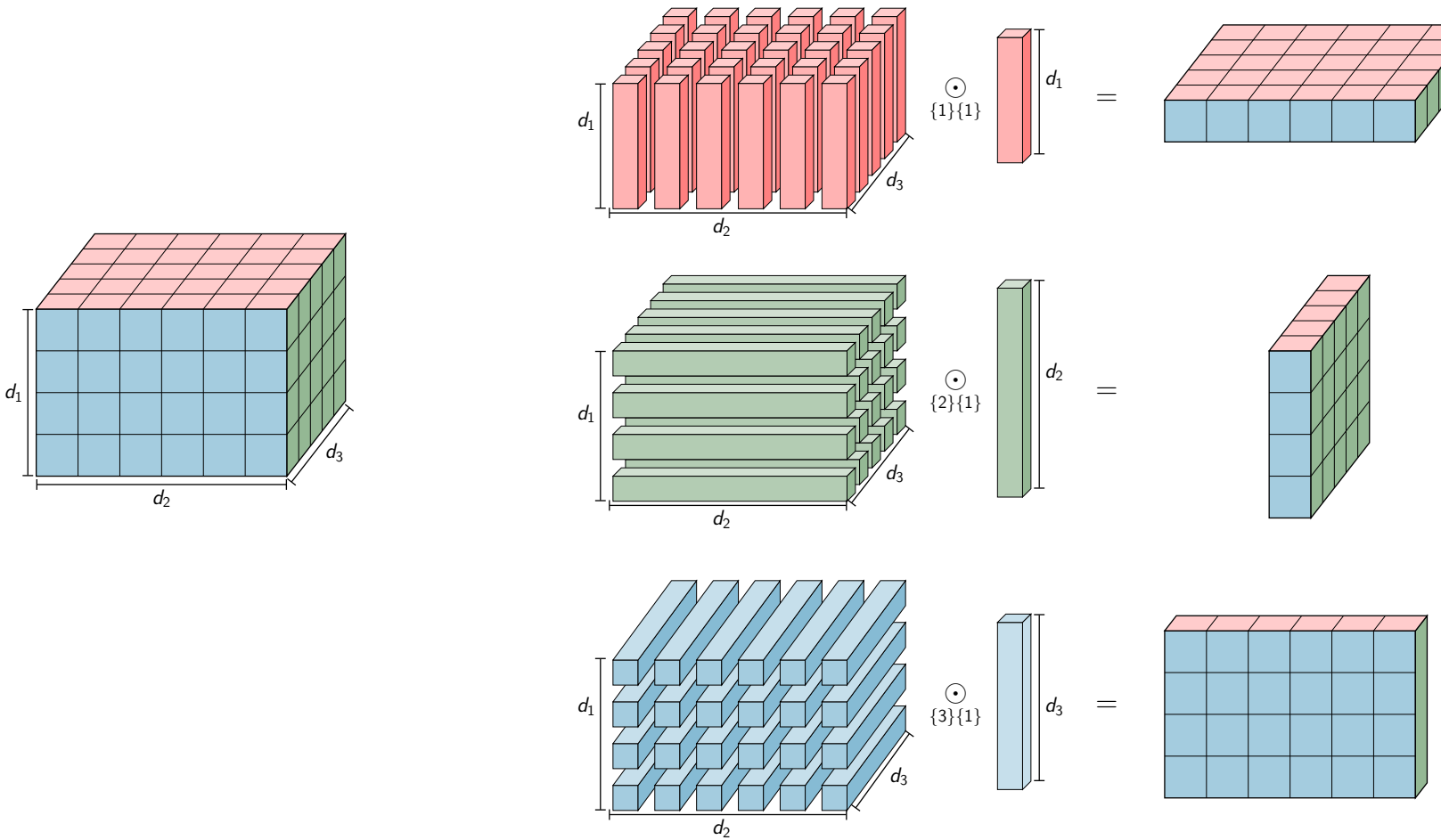
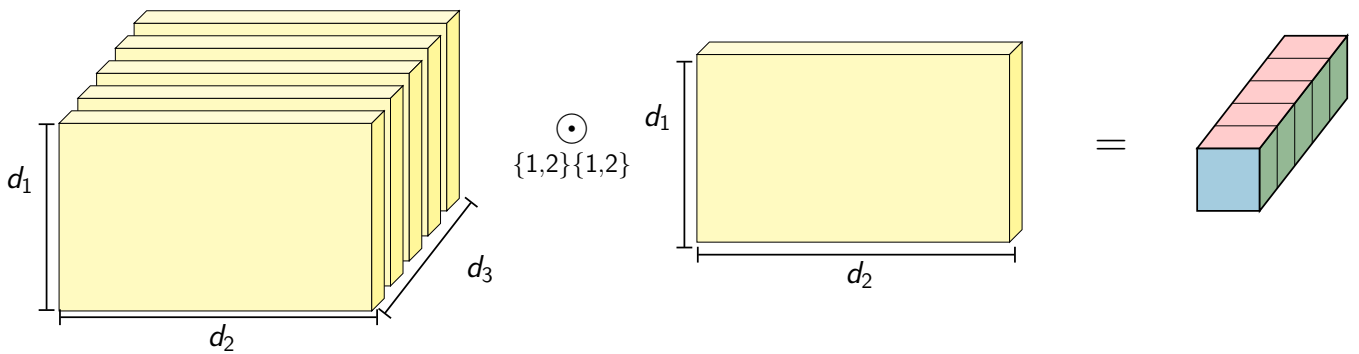


Figure: Kontraktion von T (links) mit u (oben), v (Mitte) und w (unten), siehe (1)

Allgemein werden bei Tensorkontraktion stets die Elemente entlang der kontrahierten Dimensionen multipliziert, dann aufsummiert. Somit bleiben die nicht-kontrahierten Dimensionen erhalten. Zweidimensionale Tensorkontraktion illustrieren wir anhand von $M \in \mathbb{R}^{d_1 \times d_2}$. Für T und M ist folgende Operation gültig:

$$T \underset{\{1,2\}\{1,2\}}{\odot} M = z \quad \text{wobei } z \in \mathbb{R}^{d_3} \quad \text{und} \quad z(i) = \sum_{k=0}^{d_1} \sum_{l=0}^{d_2} M(k,l) \cdot T(k,l,i)$$



Allgemein sind nur Kontraktionen gültig wenn die Größe der kontrahierten Dimensionen übereinstimmen. Zum Beispiel wäre $T \underset{\{2,3\}\{1,2\}}{\odot} M$ nicht gültig (solange $d_2 \neq d_3$).

1. Sei eine Matrix Q gegeben, sodass die Kontraktion $T \underset{\{1,3\}\{1,2\}}{\odot} Q$ gültig ist, wobei wie oben $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$. Welche Dimensionen muss dann Q haben? (Antwort in die Kommentare)

2. Ein Tensor $T \in \mathbb{R}^{2\times3\times4}$ und Matritzen $M \in \mathbb{R}^{2\times3}$, $S \in \mathbb{R}^{3\times4}$ sollen initialisiert werden. Dabei verwenden wir
   - Für den Matrix-typ std::vector<std::vector<**double**>>.
   - Für den Tensor-typ std::vector<std::vector<std::array<**double**, ?>>>. Initialisiere "?" passend als Konstante.

3. Alle Einträge von T, M und S sollen auf 1 gesetzt werden, außer deren Diagonalelemente, die auf 2 gesetzt werden. D.h.: $T(i,i,i) = 2$, $M(i,i) = 2$ und $Q(i,i,i) = 2$.

4. Implementiere in std::vector<std::vector<**double**>> tensor_vector_kont_3_1 eine eindimensionale Tensorkontraktion $T \underset{\{3\}\{1\}}{\odot} w$ für einen Tensor aus $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ und einen Vektor $w \in \mathbb{R}^{d_3}$. Hierbei soll der Elementzugriff ausschließlich mittels **Iteratoren** geschehen.

5. Implementiere in std::vector<**double**> tensor_matrix_kont_23_12 eine zweidimensionale Tensorkontraktion $T \underset{\{2,3\}\{1,2\}}{\odot} S$ für einen $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ und eine Matrix $S \in \mathbb{R}^{d_2 \times d_3}$.
   Hierbei soll der Elementzugriff ausschließlich mittels **Iteratoren** geschehen.

6. Führe mithilfe der obigen Funktionen die folgenden Kontraktionen durch
   - $Tw = T \underset{\{3\}\{1\}}{\odot} w$ (im Code ist w gegeben)
   - $TS = T \underset{\{2,3\}\{1,2\}}{\odot} S$

   und gib Tw und TS auf der Konsole aus

Tipp: Bei 5. und 6. lohnt es sich zuerst zu überlegen, wie groß die Dimensionen des Ergebnisses sind.

Abgaben

- main.cpp