

Tensor decompositions for machine learning applications

Nick Vannieuwenhoven

FWO / KU Leuven

October 9, 2017

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

6 Conclusions

Overview

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

6 Conclusions

Tensor ...

$$d = 0$$

Scalar

$$a = \square$$

$$d = 1$$

Vector

$$\mathbf{a} = \boxed{}$$

$$d = 2$$

Matrix

$$A = \boxed{}$$

Tensor ...

 $d = 0$

Scalar

 $a = \square$ $d = 1$

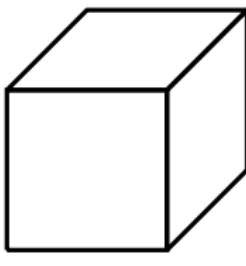
Vector

 $\mathbf{a} =$
 $d = 2$

Matrix

 $A =$
 $d \geq 3$

Tensor

 $\mathcal{A} =$


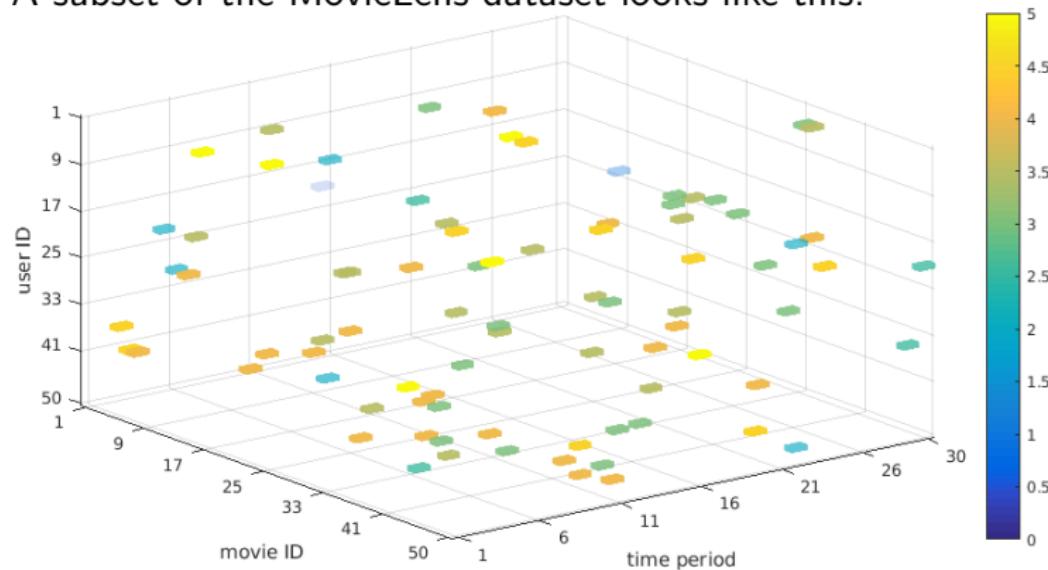


Multidimensional data naturally occurs in several machine learning problems.

In a **recommender system**, the data are (user, item, timestamp, rating) tuples. This is naturally represented as a 3rd order tensor after binning the time dimension.

In the Netflix prize challenge, for example, the winning method was an ensemble method in which many of the individual predictors exploited the temporal characteristics of the ratings.

A subset of the MovieLens dataset looks like this:



Introduction

... decompositions ...

... Decompositions ...

The **decompositional approach to matrix computations** has assumed a most prominent role in numerical linear algebra.

Of particular importance in **data analysis** applications are matrices that can be well approximated by a **low-rank matrix**.

A matrix $M \in \mathbb{R}^{m \times n}$ is of “low rank” if it can be factored as

$$M = AB^T = \sum_{i=1}^r \mathbf{a}_i \mathbf{b}_i^T$$

with r small relative to m and n , and where

$$A = [\mathbf{a}_1 \ \cdots \ \mathbf{a}_r] \in \mathbb{R}^{m \times r} \text{ and } B = [\mathbf{b}_1 \ \cdots \ \mathbf{b}_r] \in \mathbb{R}^{n \times r}.$$

Visually,

$$\boxed{M} = \boxed{A} \boxed{B^T} = \boxed{} \boxed{\mathbf{a}_1} + \dots + \boxed{} \boxed{\mathbf{a}_r}$$
$$\qquad\qquad\qquad \boxed{} \boxed{\mathbf{b}_1^T} + \dots + \boxed{} \boxed{\mathbf{b}_r^T}$$

In the machine learning literature, an expression like

$$M = AB^T \quad \text{with } A \in \mathbb{R}^{m \times r} \text{ and } B \in \mathbb{R}^{n \times r}$$

is said to reveal a **latent structure** in a data matrix M .

Some nice features of this model are:

- there are **fast algorithms** for computing low-rank matrix factorizations of complete and incomplete data;
- the decomposition is **data sparse** ($(m + n) \cdot r$ numbers versus mn); and
- easy to manipulate (esp. linear transformations).

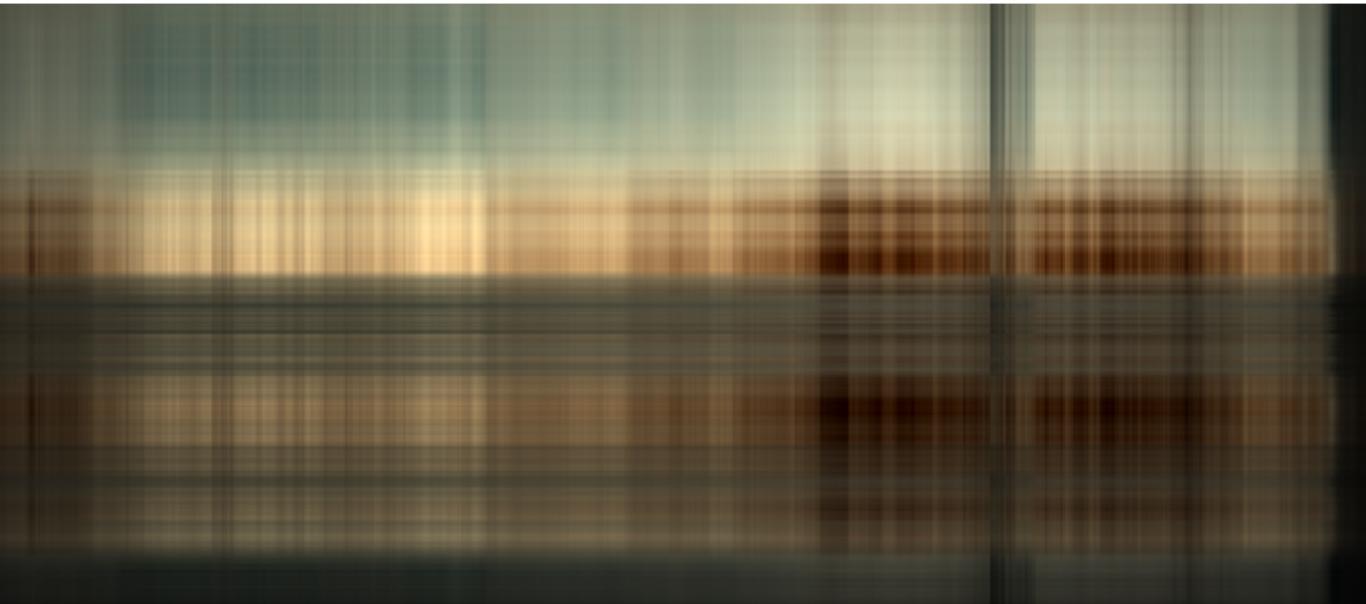
Introduction

... decompositions ...

Rank 1

Introduction

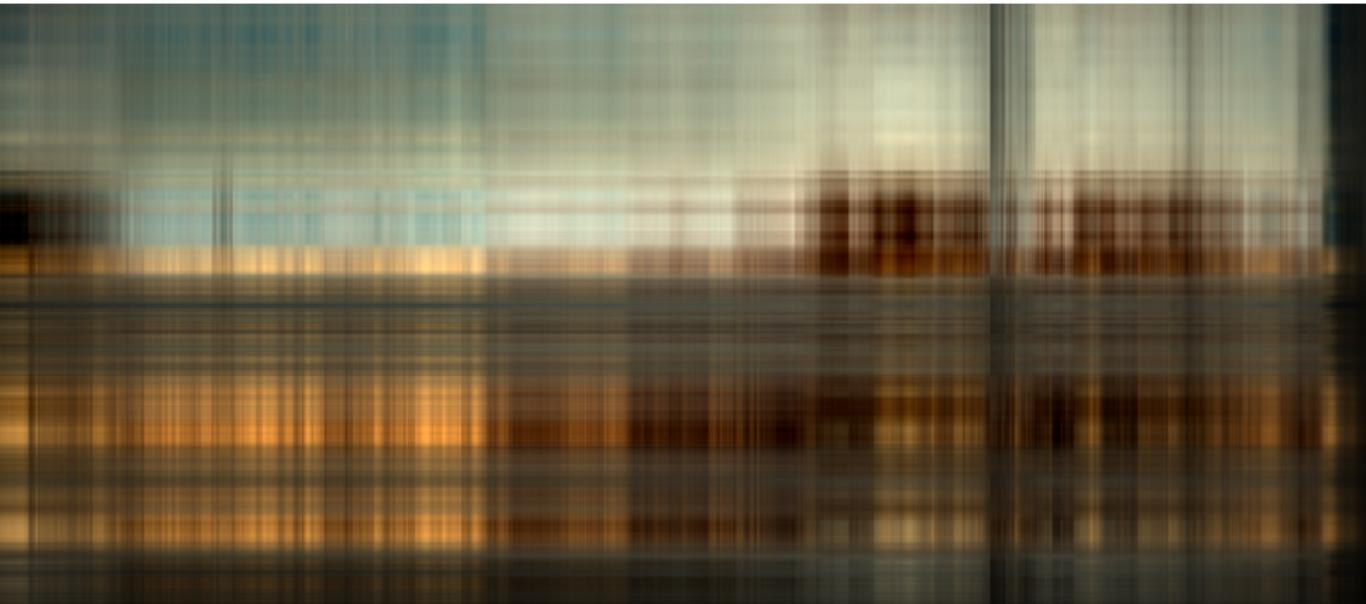
... decompositions ...



Rank 2

Introduction

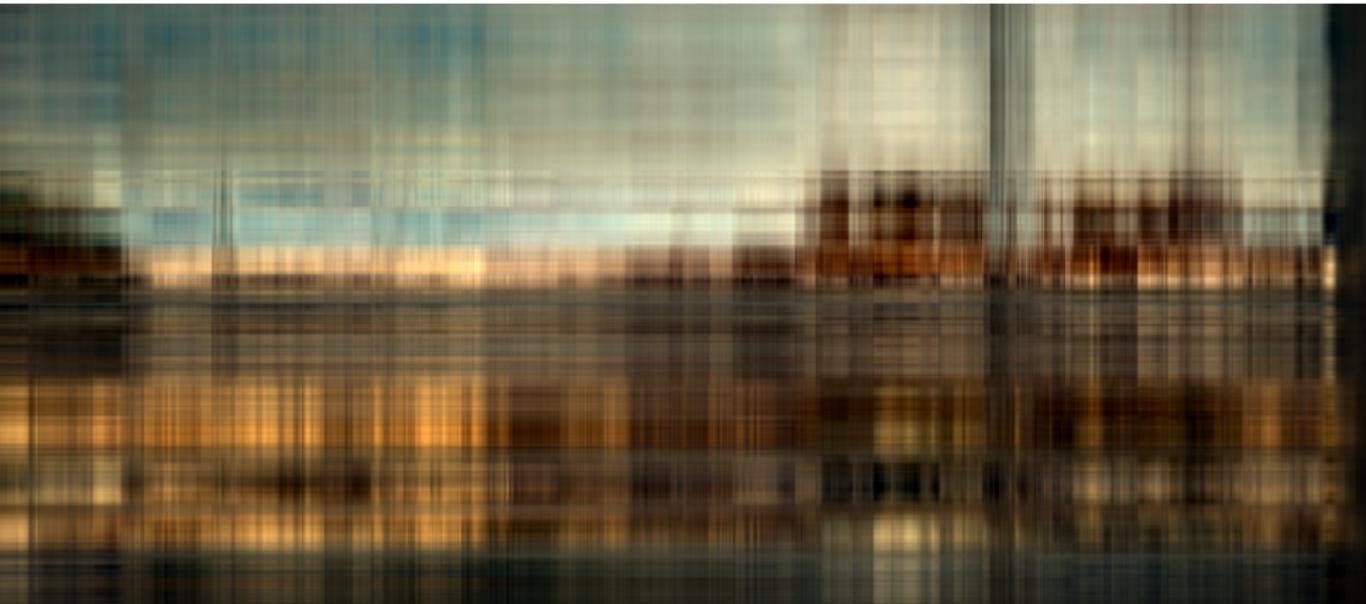
... decompositions ...



Rank 3

Introduction

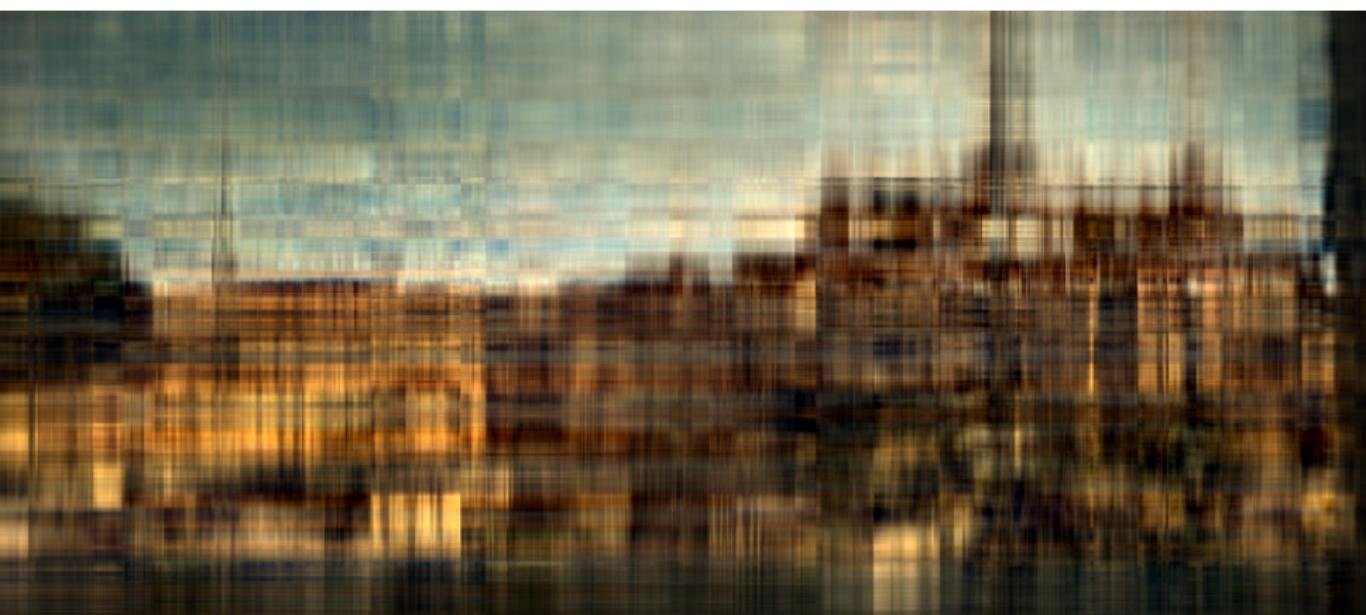
... decompositions ...



Rank 5

Introduction

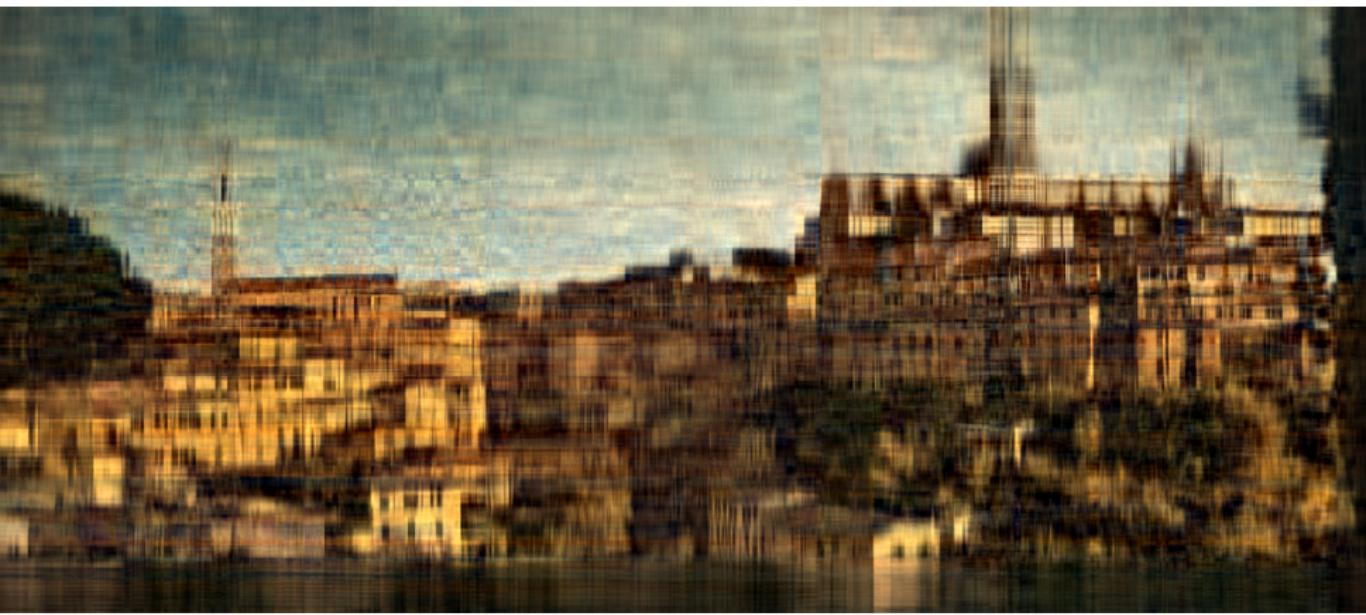
... decompositions ...



Rank 10

Introduction

... decompositions ...



Rank 25

Introduction

... decompositions ...



Rank 50

Introduction

... decompositions ...



Rank 100

Introduction

... decompositions ...



Rank 150

Introduction

... decompositions ...



Rank 200

... for machine learning applications

As far as I know, matrix factorizations have two main applications in machine learning:

- **data dimensionality reduction** (SVD), and
- **matrix completion** problems and **recommender systems** (e.g., NMF).

Unfortunately, these methods only deal naturally with 2-mode data. Nowadays, one often transforms multi-modal data to a matrix format, hereby ignoring some of the inherent structure, and then applies classic matrix methods.

A natural approach consists of applying tensor decompositions:

One anticipates good performance in scenarios where the matrix factorization framework is competitive.

Overview

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

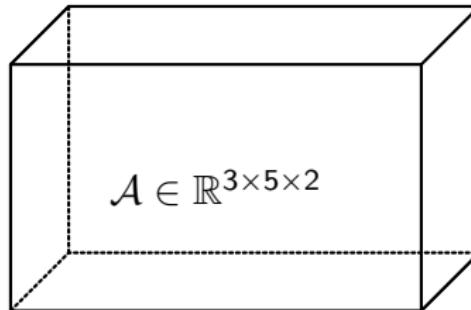
6 Conclusions

Gear 1: Vector space structure

In this talk, an **order- d tensor** of size $n_1 \times n_2 \times \cdots \times n_d$ is identified with a **multidimensional array**:

$$\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$$

Visually,



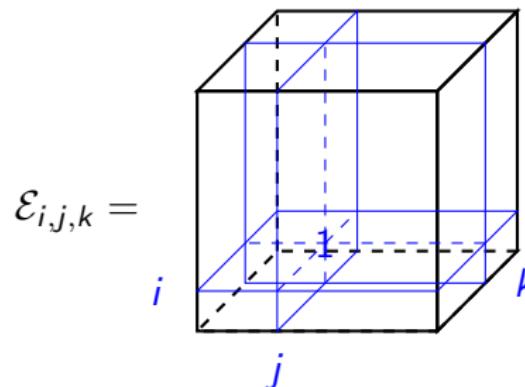
Since $\mathbb{R}^{n_1 \times \cdots \times n_d}$ is a **vector space**, we have

$$\alpha\mathcal{A} + \beta\mathcal{B} \in \mathbb{R}^{n_1 \times \cdots \times n_d} \quad \text{for all } \alpha, \beta \in \mathbb{R} \text{ and } \mathcal{A}, \mathcal{B} \in \mathbb{R}^{n_1 \times \cdots \times n_d}.$$

A **basis** for this vector space is given by the **standard tensor basis** $\{\mathcal{E}_{j_1, \dots, j_d}\}_{j_1, \dots, j_d}$, where

$$(\mathcal{E}_{j_1, \dots, j_d})_{i_1, \dots, i_d} := \begin{cases} 1 & \text{if } i_1 = j_1, i_2 = j_2, \dots, i_d = j_d, \\ 0 & \text{otherwise,} \end{cases}$$

Visually,



Every tensor

$$\mathcal{A} = [a_{i_1, \dots, i_d}]_{i_1, \dots, i_d=1}^{n_1, \dots, n_d}$$

can thus be expressed trivially as

$$\mathcal{A} = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \cdots \sum_{i_d=1}^{n_d} a_{i_1, i_2, \dots, i_d} \mathcal{E}_{i_1, i_2, \dots, i_d}.$$

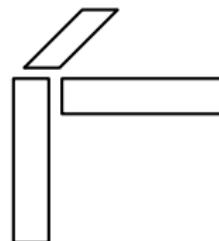
Gear 2: Tensor product of vectors

The **tensor product** of vectors $\mathbf{a}^{(k)} \in \mathbb{R}^{n_k}$, $k = 1, \dots, d$, is defined as the $n_1 \times \dots \times n_d$ multidimensional array

$$(\mathbf{a}^{(1)} \otimes \mathbf{a}^{(2)} \otimes \dots \otimes \mathbf{a}^{(d)})_{i_1, i_2, \dots, i_d} := a_{i_1}^{(1)} a_{i_2}^{(2)} \dots a_{i_d}^{(d)}.$$

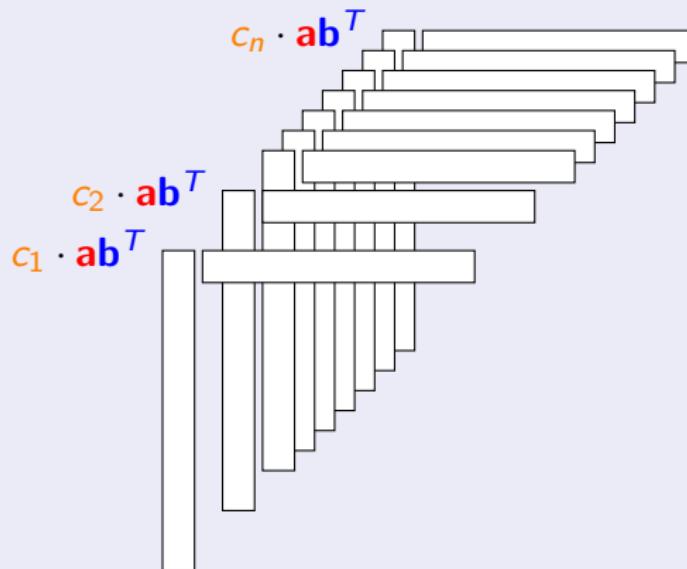
We call $\mathbf{a}^{(1)} \otimes \dots \otimes \mathbf{a}^{(d)}$ a **simple tensor**.

Visually,

$$\mathbf{a}^{(1)} \otimes \mathbf{a}^{(2)} \otimes \mathbf{a}^{(3)} =$$


For example, when $d = 2$, we have $\mathbf{a} \otimes \mathbf{b} = \mathbf{ab}^T$.

For example, when $d = 3$, we have that $\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}$ equals



Overview

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

6 Conclusions

The CP decomposition

Let $\mathbf{e}_i^{(k)} \in \mathbb{R}^{n_k}$ be the i th standard basis vector of \mathbb{R}^{n_k} .
We can then verify that

$$\mathcal{E}_{i_1, i_2, \dots, i_d} = \mathbf{e}_{i_1}^{(1)} \otimes \mathbf{e}_{i_2}^{(2)} \otimes \cdots \otimes \mathbf{e}_{i_d}^{(d)};$$

i.e., the standard tensor basis is comprised of simple tensors.

Therefore, we have

$$\mathcal{A} = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} a_{i_1, \dots, i_d} \mathbf{e}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{e}_{i_d}^{(d)},$$

which expresses a tensor as a linear combination of simple tensors.

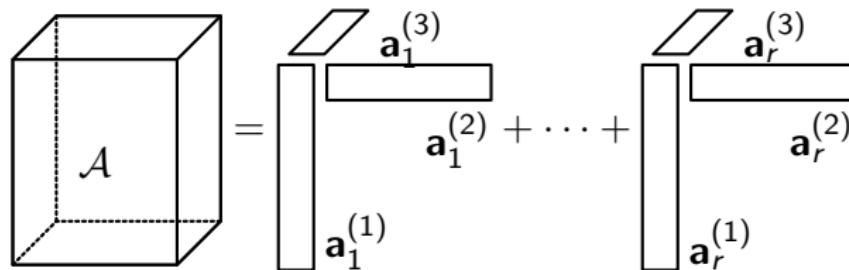
The CP decomposition

Definition

The polyadic decomposition of a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$ is an expression of the form

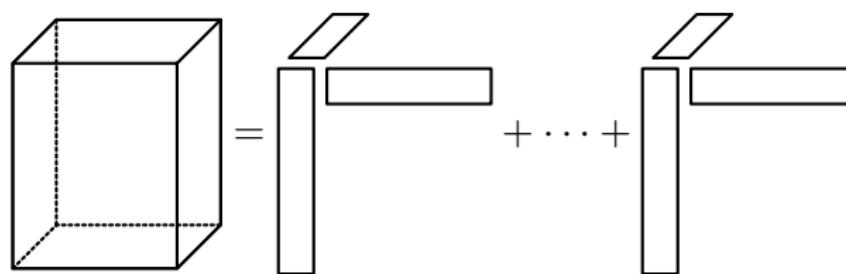
$$\mathcal{A} = \sum_{i=1}^r \mathbf{a}_i^{(1)} \otimes \mathbf{a}_i^{(2)} \otimes \dots \otimes \mathbf{a}_i^{(d)} \quad \text{with } \mathbf{a}_i^{(k)} \in \mathbb{R}^{n_k}.$$

Visually,



Expressions

$$\mathcal{A} = \sum_{i=1}^r \mathbf{a}_i^{(1)} \otimes \mathbf{a}_i^{(2)} \otimes \cdots \otimes \mathbf{a}_i^{(d)}$$



of minimal length are called **CP decompositions** or **tensor rank decompositions** of \mathcal{A} .¹

The length of a CP decomposition is called the **tensor rank**. A simple tensor has rank 1.

¹Introduced by Hitchcock (1927).

For example, for matrices ($d = 2$), a CP decomposition is

$$\mathcal{A} = \sum_{i=1}^r \mathbf{a}_i \otimes \mathbf{b}_i = \sum_{i=1}^r \mathbf{a}_i \mathbf{b}_i^T = AB^T;$$

that is, it is a low-rank matrix factorization.

The CP decomposition

Uniqueness

Uniqueness

The CP decomposition has one tremendous property:

uniqueness or identifiability

A rank-1 tensor is **essentially unique** in the sense that

$$\mathcal{A} := \mathbf{a}^{(1)} \otimes \cdots \otimes \mathbf{a}^{(d)} = \mathbf{b}^{(1)} \otimes \cdots \otimes \mathbf{b}^{(d)}$$

if and only if

$$\mathbf{a}^{(k)} = \alpha_k \mathbf{b}^{(k)} \text{ for } k = 1, \dots, d, \text{ and } \alpha_1 \cdots \alpha_d = 1.$$

A CP decomposition of a rank- r tensor \mathcal{A} is **unique** if

$$\mathcal{A} = \sum_{i=1}^r \mathcal{A}_i = \sum_{i=1}^r \mathcal{B}_i \quad \text{with } \mathcal{A}_i, \mathcal{B}_i \text{ simple tensors}$$

implies that there exists a permutation π such that

$$\mathcal{A}_i = \mathcal{B}_{\pi_i}, \quad i = 1, 2, \dots, r.$$

Uniqueness does **not** hold for rank- r matrices ($d = 2$).

Indeed, if $M = \sum_{i=1}^r \mathbf{a}_i \otimes \mathbf{b}_i$, then

$$M = AB^T = \underbrace{(AX^T)}_Y \underbrace{(BX^{-1})^T}_{Z^T} := \sum_{i=1}^r \mathbf{y}_i \otimes \mathbf{z}_i$$

such that for a generic invertible matrix $X \in \mathbb{R}^{r \times r}$ we have $\alpha_i \mathbf{y}_i \neq \mathbf{a}_{\pi_i}$ for all $i = 1, \dots, r$, all permutations π , and all $\alpha_i \in \mathbb{R}$.

In the general case, $d \geq 3$, it is conjectured by that the generic rank- r tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ has a unique CP decomposition if

$$r < r_c := \frac{\prod_{k=1}^d n_k}{1 + \sum_{k=1}^d (n_k - 1)},$$

unless it is one of the exceptional cases listed in Theorem 1.1 of Chiantini, Ottaviani and V (2014).

This conjecture is true for all $\prod_{k=1}^d n_k \leq 15000$.

If $r > r_c$, then there are always ∞ -many CP decompositions of \mathcal{A} .

Algorithms

Direct algorithms for computing a rank- r CP decomposition exist only for small rank. They are based on a **generalized eigenvalue decomposition**.

In general, **optimization methods** are applied to minimize

$$\frac{1}{2} \left\| \mathcal{A} - \sum_{i=1}^r \mathbf{a}_i^{(1)} \otimes \cdots \otimes \mathbf{a}_i^{(d)} \right\|_F^2,$$

possibly with regularization. Two popular strategies are:

- alternating least squares (ALS), or block coordinate descent; and
- structure-exploiting nonlinear least-squares solvers (e.g., Gauss-Newton methods).

Application in machine learning

Consider **document clustering** in the unsupervised learning setting using an **exchangeable single topic model**.²

There are three assumptions in this model:

- ① each document is described by a **bag of words**, i.e., order is irrelevant;
- ② the documents contain r distinct topics and the **probability of words is conditional on the topic** of the document;
- ③ the words W in a document are **independently and identically distributed** (i.i.d.) conditional on the topic.

²The theory was developed in Allman, Matias and Rhodes (2009), and Anandkumar et al. (2014).

With these assumptions, the total probability distribution is

$$P(W) = \sum_{\ell=1}^r P(T = t_\ell) \cdot P(W | T = t_\ell).$$

Let's use the more compact notation

$$\mathbf{p} = \sum_{\ell=1}^r \alpha_\ell \mathbf{p}_\ell,$$

where $\alpha_\ell := P(T = t_\ell)$.

Statistical parameter inference problem

How do we find the probability distribution vectors \mathbf{p}_ℓ for each topic t_ℓ ? How do we identify the parameters of the model?

All that we are handed is:

- ① A set of k documents D ,
- ② (A set of n key words W), and
- ③ (The number of topics r).

We assume (2) and (3) for convenience.

Clearly, we can compute a sample statistic $\hat{\mathbf{p}}$ for estimating \mathbf{p} by counting word frequencies.

However, we cannot recover the conditional probabilities from $\hat{\mathbf{p}} \approx \mathbf{p}$, because

vectors are not identifiable!

Recall Assumption 3: the distribution of the words is *i.i.d.*, *conditional on the topic*. Therefore,

$$\begin{aligned} P(W = w_i, W = w_j) &:= \sum_{\ell=1}^r \alpha_\ell \cdot P(W = w_i, W = w_j \mid T = t_\ell) \\ &= \sum_{\ell=1}^r \alpha_\ell \cdot P(W = w_i \mid T = t_l) P(W = w_j \mid T = t_\ell) \end{aligned}$$

so that we get

$$P := \sum_{\ell=1}^r \alpha_\ell \cdot \mathbf{p}_\ell \otimes \mathbf{p}_\ell$$

This is a **rank- r symmetric matrix decomposition!**

Again, we can compute a sample statistic \hat{P} that approximates the true P by counting the frequency of word pairs.

However, stochastic matrices are not identifiable!

Again, we can compute a sample statistic \hat{P} that approximates the true P by counting the frequency of word pairs.

However, stochastic matrices are not identifiable!

Indeed, consider the map

$$\mathbb{R}^{r-1} \times (\mathbb{R}^{n-1})^{\times r} \rightarrow \mathbb{R}^{n \times n},$$

$$([\alpha_i]_{i=1}^{r-1}, \hat{\mathbf{p}}_1, \dots, \hat{\mathbf{p}}_r) \mapsto \sum_{\ell=1}^r \alpha_\ell \begin{bmatrix} \hat{\mathbf{p}}_\ell \\ 1 - \mathbf{1}^T \hat{\mathbf{p}}_\ell \end{bmatrix} \begin{bmatrix} \hat{\mathbf{p}}_\ell & 1 - \mathbf{1}^T \hat{\mathbf{p}}_\ell \end{bmatrix},$$

where $\alpha_r := 1 - \sum_{\ell=1}^{r-1} \alpha_\ell$.

The source has dimension $r - 1 + r(n - 1)$, but the image is contained in the set of symmetric matrices of rank bounded by r , which only has dimension $\binom{n+1}{2} - \binom{n+1-r}{2}$. For $r \geq 2$ the latter is smaller than the former. It follows from the fiber dimension theorem (and the fact that the map above is a polynomial map with an \mathbb{R} -variety as source) that every fiber has a positive-dimensional component.

What about word **triplets**?

We have

$$P(W = w_i, W = w_j, W = w_k)$$

$$= \sum_{\ell=1}^r \alpha_\ell P(W = w_i \mid T = t_\ell) P(W = w_j \mid T = t_\ell) P(W = w_k \mid T = t_\ell)$$

so that we get

$$\mathcal{P} := \sum_{\ell=1}^r \alpha_\ell \cdot \mathbf{p}_\ell \otimes \mathbf{p}_\ell \otimes \mathbf{p}_\ell.$$

This is a (symmetric) **rank- r CP decomposition!**

Chiantini, Ottaviani, and V (2017) proved that generic low-rank symmetric CP decompositions are unique. Hence,

the conditional probability distributions \mathbf{p}_i can be recovered from the unique rank- r CP decomposition of $\widehat{\mathcal{P}} \approx \mathcal{P}$.

The final stage consists of classifying the k documents by exploiting the statistical model.

We choose the topic that maximizes the a posteriori likelihood of the observed key words in the document.

Allman, Matias and Rhodes (2009) and Anandkumar et al. (2014) showed that the parameters of several **latent variable models**, including

- single topic models,
- spherical Gaussian mixtures,
- latent Dirichlet allocation (LDA),
- naive Bayes models, and
- hidden Markov models (HMMs)

can be recovered exactly from computing the CP decomposition of an appropriate moment tensor.

Overview

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

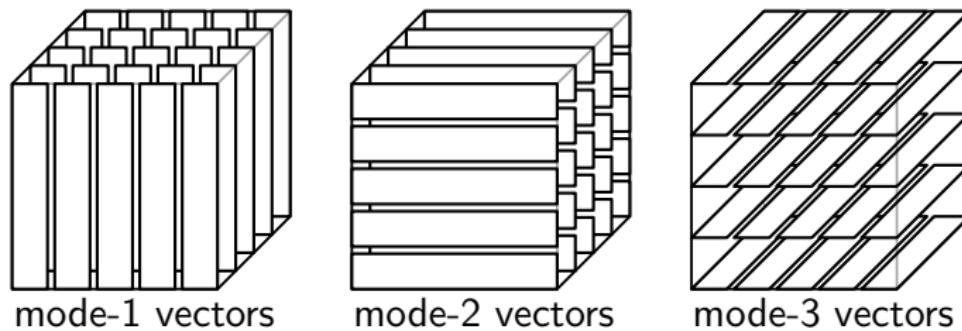
5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

6 Conclusions

Gear 1: The mode- k vector space

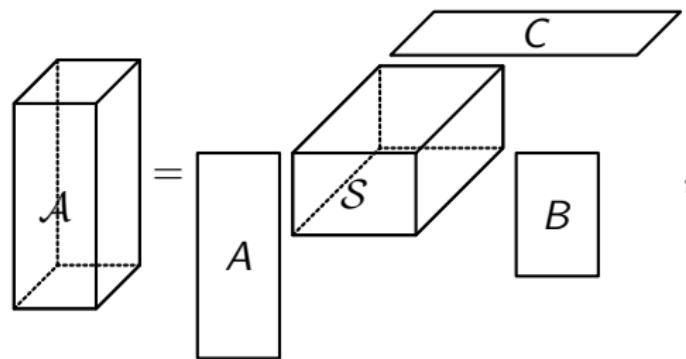
We can associate d vector spaces with a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$.



The space spanned by the mode- k vectors is called the **mode- k vector space** associated with \mathcal{A} ; we denote it by A_k .

Gear 3: Tensor product of operators

A tensor \mathcal{S} can be multiplied “on all sides” by matrices. This is called a **multilinear multiplication**.



where $\mathcal{A} \in \mathbb{R}^{m_1 \times m_2 \times m_3}$, $\mathcal{S} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $A \in \mathbb{R}^{m_1 \times n_1}$, $B \in \mathbb{R}^{m_2 \times n_2}$, and $C \in \mathbb{R}^{m_3 \times n_3}$.

Let

- ① a simple tensor $\mathcal{A} = \mathbf{a}^{(1)} \otimes \cdots \otimes \mathbf{a}^{(d)} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$; and
- ② a collection of matrices $M_k \in \mathbb{R}^{m_k \times n_k}$, $k = 1, \dots, d$.

Multilinear multiplication is **defined for simple tensors** as

$$(M_1, M_2, \dots, M_d) \cdot \mathcal{A} := (M_1 \mathbf{a}^{(1)}) \otimes (M_2 \mathbf{a}^{(2)}) \otimes \cdots \otimes (M_d \mathbf{a}^{(d)});$$

this tensor lives in $\mathbb{R}^{m_1 \times m_2 \times \cdots \times m_d}$.

Multilinear multiplication is a **linear operator**:

$$(M_1, \dots, M_d) \cdot (\alpha \mathcal{A} + \beta \mathcal{B}) = \alpha(M_1, \dots, M_d) \cdot \mathcal{A} + \beta(M_1, \dots, M_d) \cdot \mathcal{B}.$$

Linearity and the definition for simple tensors suffices for the general case, since

$$\begin{aligned}
 \mathcal{A} &= (M_1, \dots, M_d) \cdot \mathcal{S} \\
 &= (M_1, \dots, M_d) \cdot \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} s_{i_1, \dots, i_d} \mathbf{e}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{e}_{i_d}^{(d)} \\
 &= \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} s_{i_1, \dots, i_d} (M_1, \dots, M_d) \cdot (\mathbf{e}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{e}_{i_d}^{(d)}) \\
 &= \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} s_{i_1, \dots, i_d} (M_1 \mathbf{e}_{i_1}^{(d)}) \otimes \cdots \otimes (M_d \mathbf{e}_{i_d}^{(d)}) \\
 &= \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} s_{i_1, \dots, i_d} \mathbf{m}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{m}_{i_d}^{(d)},
 \end{aligned}$$

where $M_k := [\mathbf{m}_i^{(k)}]_{i=1}^{m_k}$.

As an example multilinear multiplication, consider $d = 2$. Let

$$S = AB^T = \sum_{i=1}^r \mathbf{a}_i \mathbf{b}_i^T = \sum_{i=1}^r \mathbf{a}_i \otimes \mathbf{b}_i.$$

Then, we have

$$\begin{aligned} (M_1, M_2) \cdot S &:= (M_1, M_2) \cdot \sum_{i=1}^r \mathbf{a}_i \otimes \mathbf{b}_i && \text{(definition of } A\text{)} \\ &= \sum_{i=1}^r (M_1, M_2) \cdot (\mathbf{a}_i \otimes \mathbf{b}_i) && \text{(linearity)} \\ &= \sum_{i=1}^r (M_1 \mathbf{a}_i) \otimes (M_2 \mathbf{b}_i) && \text{(definition)} \\ &= (M_1 A)(M_2 B)^T \\ &= M_1 S M_2^T. \end{aligned}$$

Overview

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

6 Conclusions

The Tucker decomposition

The **orthogonal Tucker decomposition**³ of a tensor

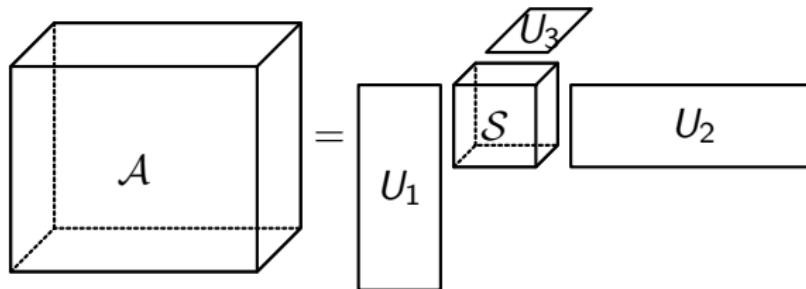
$\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$ is an expression of the form

$$\mathcal{A} = (U_1, U_2, \dots, U_d) \cdot \mathcal{S},$$

where $\mathcal{S} \in \mathbb{R}^{r_1 \times r_2 \times \dots \times r_d}$, and $U_k \in \mathbb{R}^{n_k \times r_k}$ contains an orthonormal basis of the mode- k vector space associated with \mathcal{A} ; hence,

$$r_k := \dim(\mathcal{A}_k) \leq n_k.$$

Visually,



³It was introduced by Tucker (1963).

The Tucker decomposition

Definition

For example, for matrices ($d = 2$), an orthogonal Tucker decomposition is

$$\mathcal{A} = (U_1, U_2) \cdot S = U_1 S U_2^T = (U_1 S) U_2^T;$$

that is, it is a [low-rank matrix factorization](#).

In fact, if $S = UDV^T$ is a singular value decomposition, then

$$\mathcal{A} = U_1 S U_2^T = (U_1 U) D (U_2 V)^T = (U_1 U, U_2 V) \cdot D$$

is another orthogonal Tucker decomposition of \mathcal{A} .

The Tucker decomposition

Definition

Since

$$\mathcal{A} = (U_1, \dots, U_d) \cdot \mathcal{S} = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} s_{i_1, \dots, i_d} \mathbf{u}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{u}_{i_d}^{(d)}$$

the orthogonal Tucker decomposition has a nice **geometric interpretation**: \mathcal{A} is expressed in the tensor basis

$\{\mathbf{u}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{u}_{i_d}^{(d)}\}_{i_1, \dots, i_d}$ by the $r_1 \cdots r_d$ coefficient array
 $\mathcal{S} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$.

The Tucker decomposition

Definition

Since

$$\mathcal{A} = (U_1, \dots, U_d) \cdot \mathcal{S} = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} s_{i_1, \dots, i_d} \mathbf{u}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{u}_{i_d}^{(d)}$$

the orthogonal Tucker decomposition has a nice **geometric interpretation**: \mathcal{A} is expressed in the tensor basis

$\{\mathbf{u}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{u}_{i_d}^{(d)}\}_{i_1, \dots, i_d}$ by the $r_1 \cdots r_d$ coefficient array
 $\mathcal{S} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$.

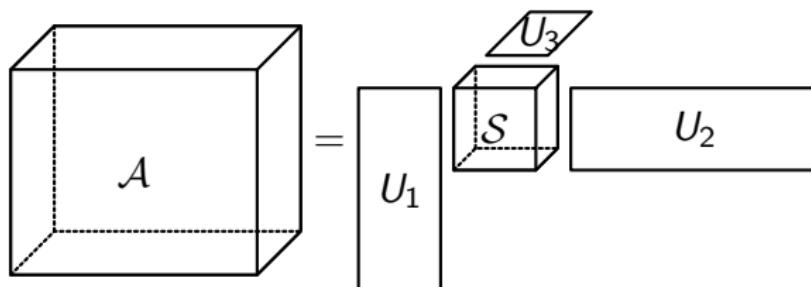
This is completely analogous to the matrix case!

A geometric interpretation of the SVD $A = USV^T$ is that every rank- r matrix A is represented as an $r \times r$ diagonal matrix S with respect to well-chosen bases for its column and row space, i.e., by the columns of U and V , respectively.

The Tucker decomposition

Definition

The orthogonal Tucker decomposition



can be thought of as revealing latent features U_k for the k th mode. The coefficient s_{i_1, \dots, i_d} in S signifies the relative importance of the “combined” feature

$$\mathbf{u}_{i_1}^{(1)} \otimes \cdots \otimes \mathbf{u}_{i_d}^{(d)}.$$

Algorithms

A Tucker decomposition

$$\mathcal{A} = (U_1, U_2, \dots, U_d) \cdot \mathcal{S}$$

can be computed by constructing orthonormal bases U_k for the mode- k vector spaces A_k , and then computing the **projection**

$$\mathcal{S} := (U_1^T, U_2^T, \dots, U_d^T) \cdot \mathcal{A}.$$

For approximating a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$ by one of multilinear rank (r_1, r_2, \dots, r_d) , the **truncated HOSVD**⁴ can be employed:

- ① Compute the dominant r_k left singular vectors U_k of A_k for $k = 1, 2, \dots, d$; and then
- ② project: $\mathcal{S} := (U_1^T, U_2^T, \dots, U_d^T) \cdot \mathcal{A}$.

The resulting approximation is **quasi-optimal**:

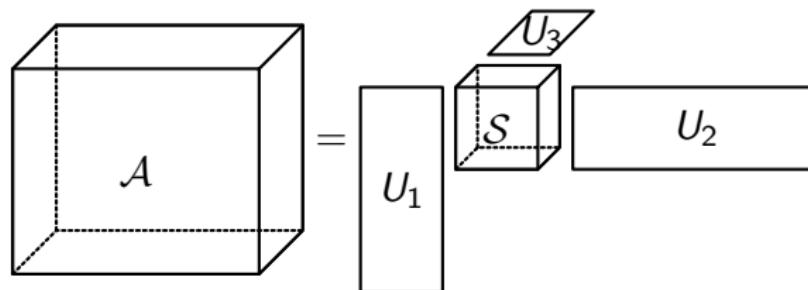
$$\|\mathcal{A} - (U_1, \dots, U_d) \cdot \mathcal{S}\| \leq \sqrt{d-1} \|\mathcal{A} - \mathcal{A}_{(r_1, \dots, r_d)}^*\|,$$

where $\mathcal{A}_{(r_1, \dots, r_d)}^*$ is the best approximation of multilinear rank (r_1, \dots, r_d) to \mathcal{A} .

⁴See De Lathauwer, De Moor and Vandewalle (2001); a faster, sequential truncation method was proposed in V, Vandebril and Meerbergen (2012).

For these reasons, the orthogonal Tucker decomposition is a great tool for **multidimensional data dimensionality reduction**.

The storage cost of



with $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$ and $\mathcal{S} \in \mathbb{R}^{r_1 \times r_2 \times \dots \times r_d}$ is, in practice,

$$\prod_{k=1}^d r_k + \sum_{k=1}^d n_k r_k.$$

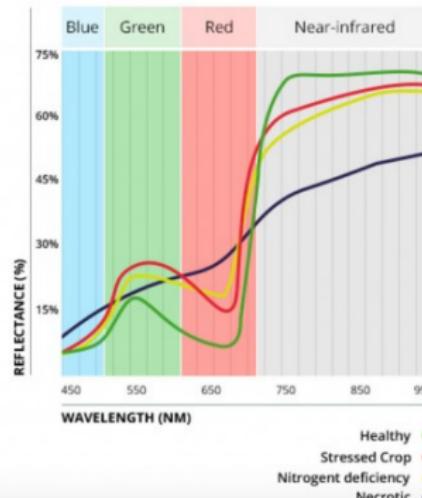
Application: data dimensionality reduction

The Swiss startup *Gamaya*⁵ for instance offers hyperspectral imaging solutions for **crop growth monitoring**.

Measure reflectance of your crop using proprietary hyperspectral imaging camera mounted on drones or manned aircrafts

Analyze spectrum of reflected light and correlate it with crop and soil characteristics

Identify potential problems of your farmland (diseases, nutrient deficiencies, weeds, environmental stresses)



We can model this as a **classification problem**: to each spatial coordinate (i, j) we want to assign a label $\ell \in \{1, \dots, k\}$ indicating which material (e.g. healthy crop, bare soil, water, etc) is visible.

The straightforward approach would take the spectral response of each pixel (i.e., a vector of length 224) and feed this to a classification method.

On the other hand, we could first apply a dimensionality reduction along the third mode, and feed instead the feature-vector automatically detected by an orthogonal Tucker decomposition.

Let's take the Salinas data set from

[http://www.ehu.eus/ccwintco/index.php?title=](http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes)

Hyperspectral_Remote_Sensing_Scenes, which contains a tiny sample (about 27 MB).

*This scene was collected by the **224-band AVIRIS sensor** over **Salinas Valley, California**, and is characterized by high spatial resolution (**3.7-meter pixels**). The area covered comprises 512 lines by 217 samples. We discarded the 20 water absorption bands, in this case bands: [108-112], [154-167], 224. This image [...] includes vegetables, bare soils, and vineyard fields.*

The Tucker decomposition

Application: data dimensionality reduction

We applied an orthogonal Tucker compression to the $512 \times 217 \times 224$ data array.

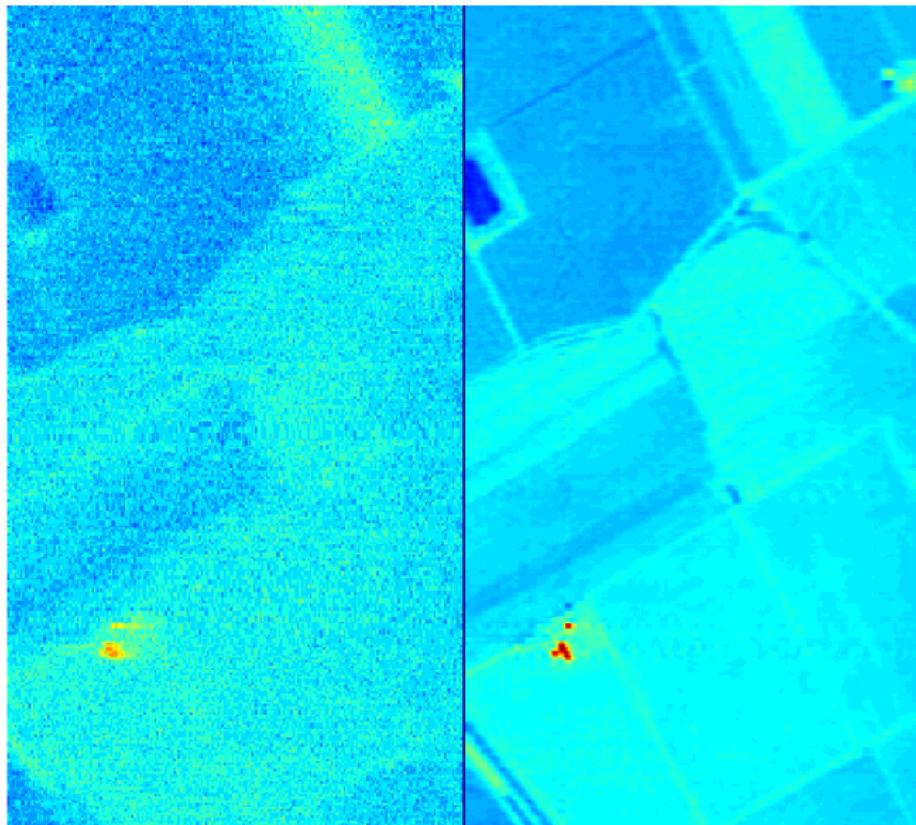
The chosen approximation had multilinear rank $(124, 50, 5)$, resulting in a relative approximation error of $5 \cdot 10^{-2}$.

The compression factor is 233.8.

Tensor decompositions for machine learning applications

The Tucker decomposition

Application: data dimensionality reduction

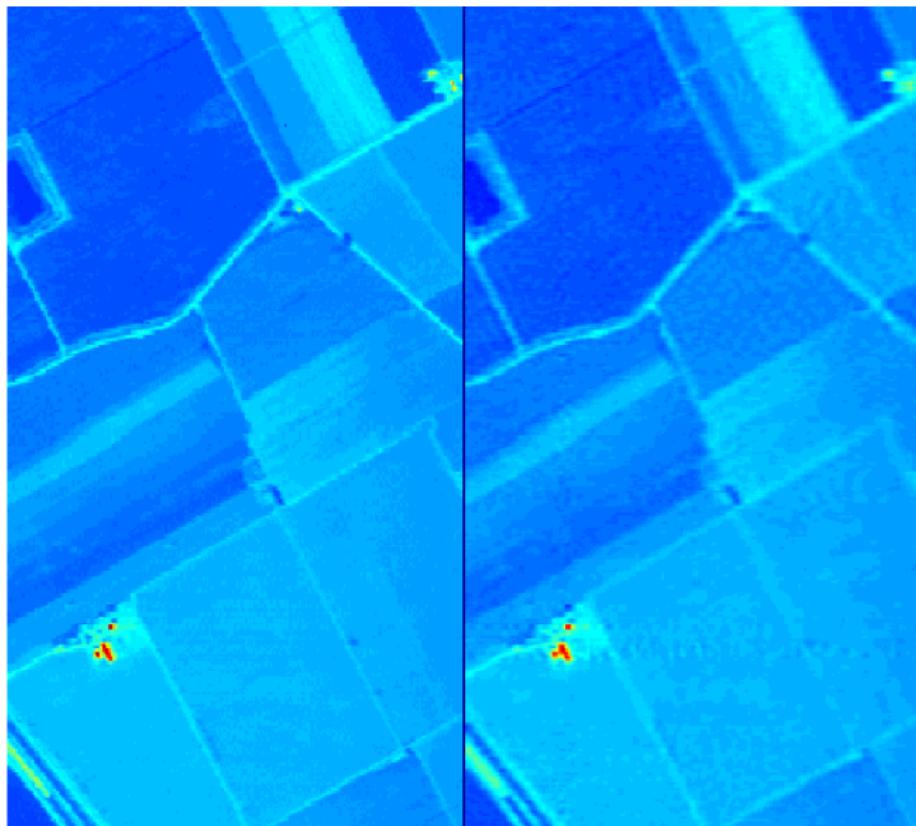


Spectral band 001

Tensor decompositions for machine learning applications

The Tucker decomposition

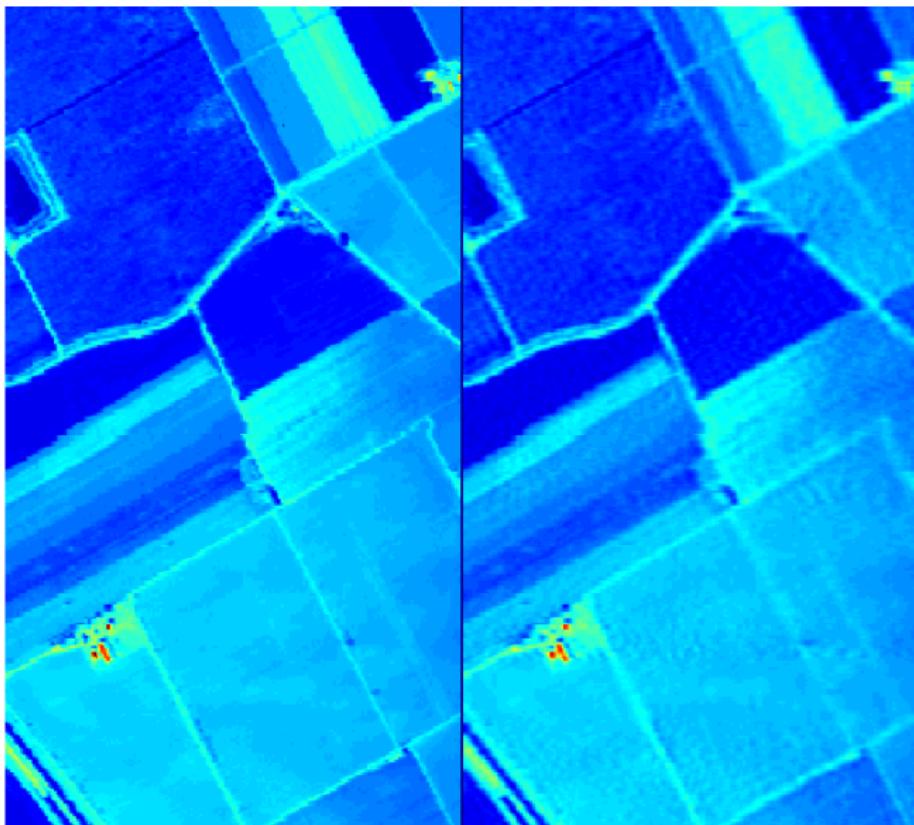
Application: data dimensionality reduction



Spectral band 007

The Tucker decomposition

Application: data dimensionality reduction

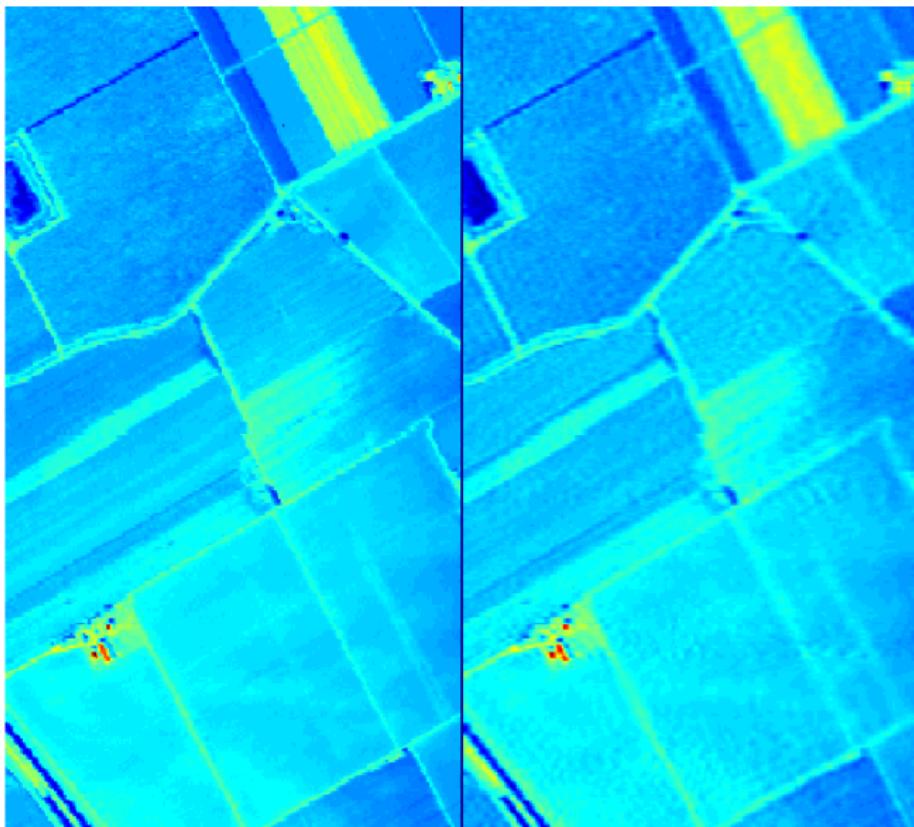


Spectral band 028

Tensor decompositions for machine learning applications

The Tucker decomposition

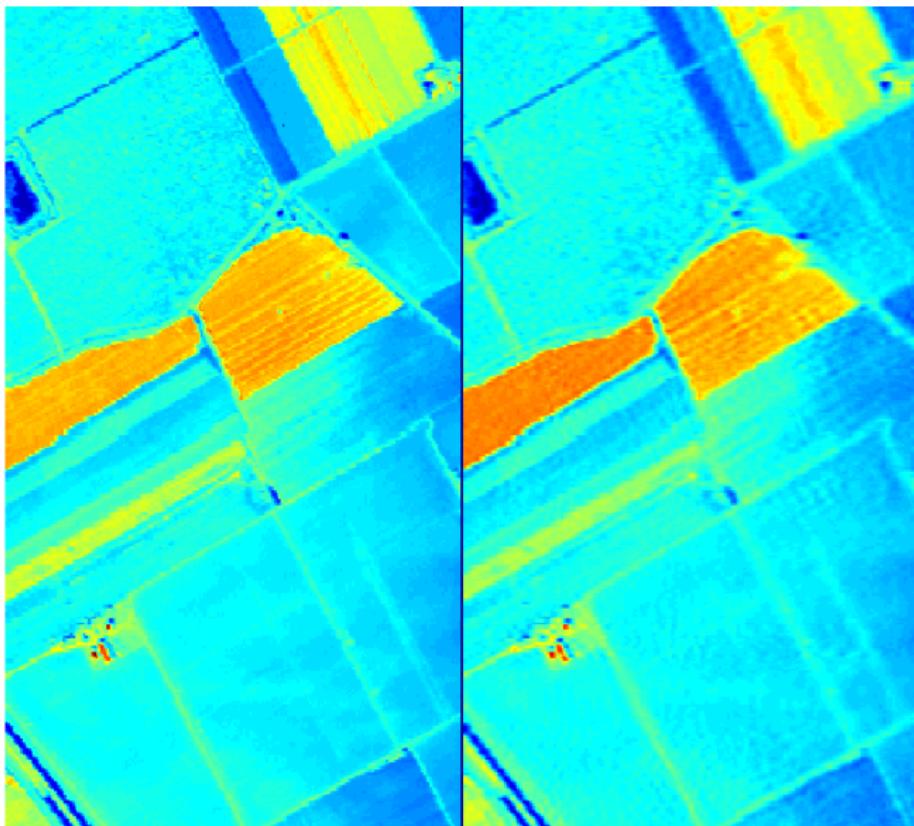
Application: data dimensionality reduction



Spectral band 038

The Tucker decomposition

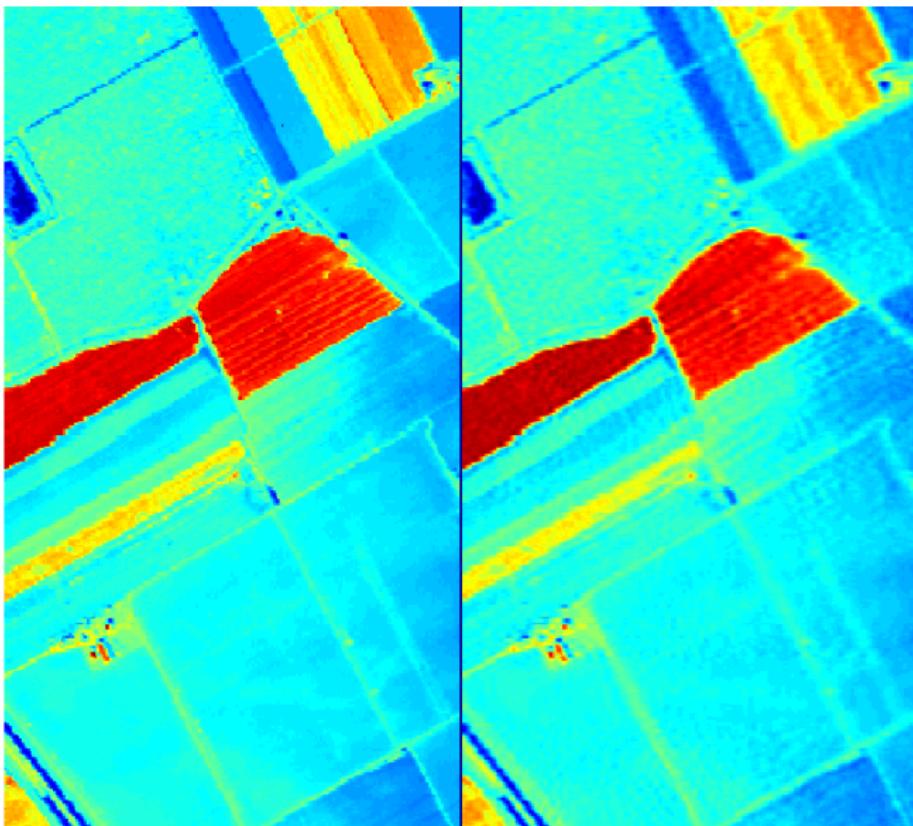
Application: data dimensionality reduction



Spectral band 040

The Tucker decomposition

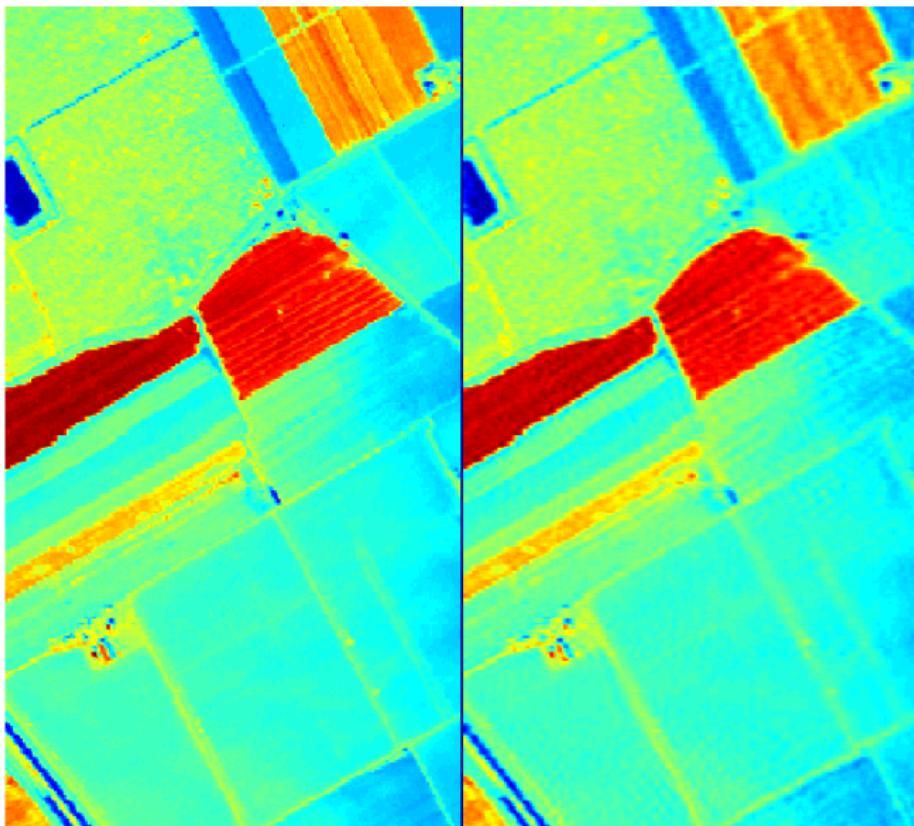
Application: data dimensionality reduction



Spectral band 041

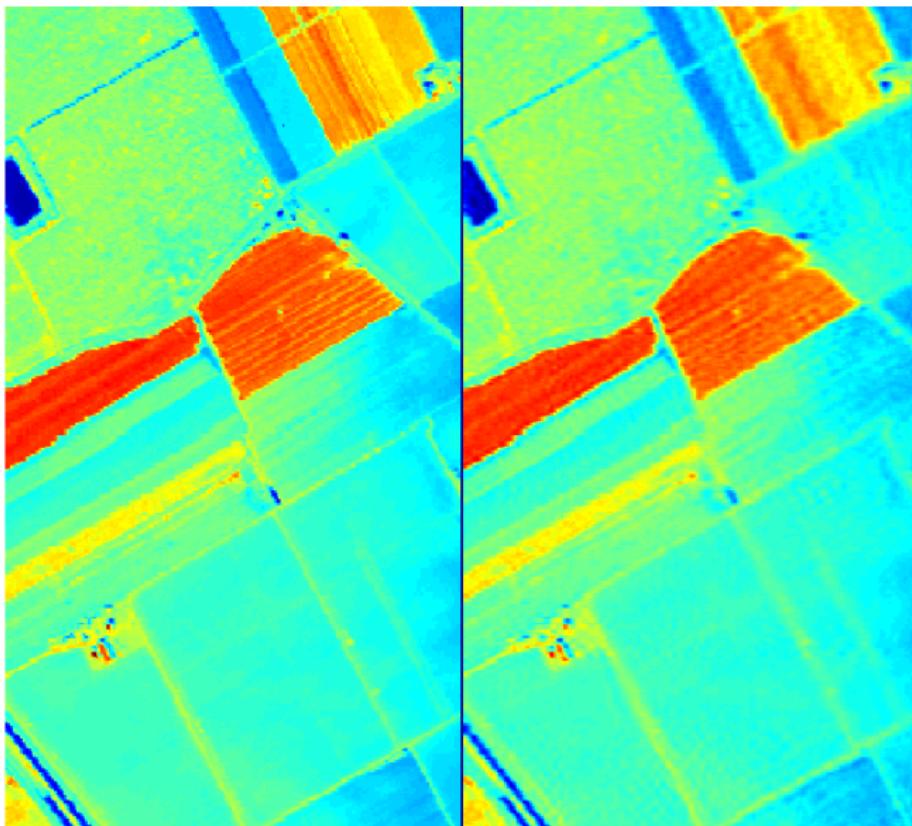
The Tucker decomposition

Application: data dimensionality reduction



The Tucker decomposition

Application: data dimensionality reduction

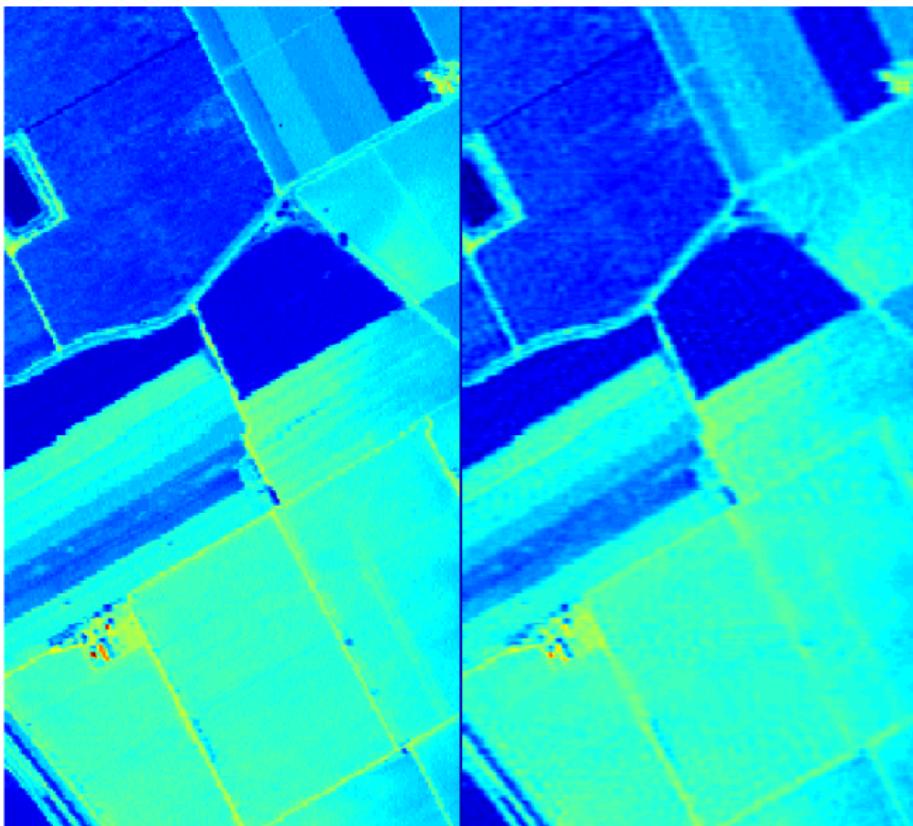


Spectral band 065

Tensor decompositions for machine learning applications

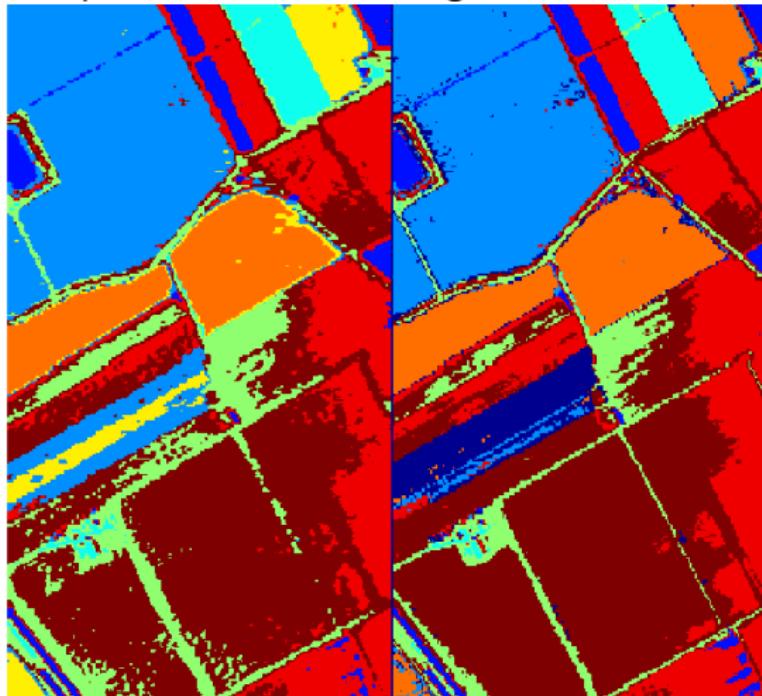
The Tucker decomposition

Application: data dimensionality reduction



Spectral band 205

We applied a simple k -means clustering with 10 clusters.



In the compressed case (left), the feature vectors were of length 5, in the uncompressed case (right) they were 224-dimensional vectors. This is a compression rate of 44!

Overview

1 Introduction

- Tensor ...
- ... decompositions ...
- ... for machine learning applications

2 Tensors and multilinear algebra I

3 The CP decomposition

- Definition
- Uniqueness
- Application: Topic modeling

4 Tensors and multilinear algebra II

5 The Tucker decomposition

- Definition
- Algorithms
- Application: data dimensionality reduction

6 Conclusions

Conclusions

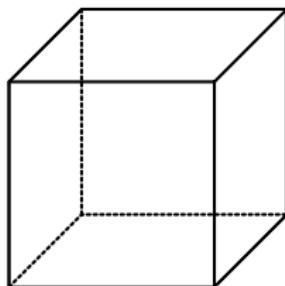
We presented the two key tensor decompositions with applications in machine learning.

The **CP decomposition**:

- is usually **unique** and hence easy to interpret;
- can be applied in recommender systems; and
- can learn the parameters of some **latent variable models**.

The **orthogonal Tucker decomposition**:

- is easy to compute;
- can reveal latent features in each mode;
- is very well-suited for **data dimensionality reduction**;
- has been applied for recommender system tasks and compression of deep neural networks.



Thanks for your attention!

References

- Greub (1978), Multilinear Algebra, Springer.
- Landsberg (2012), Tensors: Geometry and Applications, AMS.
- Hitchcock (1927), The expression of a tensor or a polyadic as a sum of products, *J. Math. Phys.*
- Chiantini, Ottaviani, — (2014), An algorithm for generic identifiability of complex tensors of subgeneric rank, *SIAM J. Matrix Anal. Appl.*
- Allman, Matias, Rhodes (2009), Identifiability of parameters in latent structure models with many observed variables, *Ann. Statist.*
- Anandkumar, Ge, Hsu, Kakade, Telgarsky (2014), Tensor decompositions for learning latent variable models, *J. Mach. Learn. Res.*
- Chiantini, Ottaviani, — (2017), On generic identifiability of symmetric tensors of subgeneric rank, *Trans. Amer. Math. Soc.*
- Tucker (1966), Some mathematical notes on three-mode factor analysis, *Psychometrika*.
- De Lathauwer, De Moor, Vandewalle (2000), A multilinear singular value decomposition, *SIAM J. Matrix Anal. Appl.*
- —, Vandebril, Meerbergen (2012), A new truncation strategy for the higher-order singular value decomposition, *SIAM J. Sci. Comput.*