Numerical tensor methods and their applications

I.V. Oseledets

7 May 2013

All lectures

4 lectures,

- 2 May, 08:00 - 10:00: Introduction: ideas, matrix results, history.
- 7 May, 08:00 - 10:00: Novel tensor formats (TT, HT, QTT).
- 8 May, 08:00 - 10:00: Advanced tensor methods (eigenproblems, linear systems).
- 14 May, 08:00 - 10:00: Advanced topics, recent results and open problems.

Brief recap of Lecture 1

Previous lecture:

Brief recap of Lecture 1

Previous lecture:

- SVD and skeleton decompositions

Brief recap of Lecture 1

Previous lecture:

- SVD and skeleton decompositions
- A tensor is a d-way array: $A(i_1, \ldots, i_d)$

Brief recap of Lecture 1

Previous lecture:

- SVD and skeleton decompositions
- A tensor is a d-way array: $A(i_1, \ldots, i_d)$
- Key idea: separation of variables

Two classical formats

Two classical formats:

- The canonical format
- The Tucker format

The canonical format

Canonical format

$$A(i_1, \ldots, i_d) = \sum_{\alpha=1}^{r} U_1(i_1, \alpha) \ldots U_d(i_d, \alpha)$$

- *dnr* parameters (low!)
- No robust algorithms
- Uniqueness, important as a data model

Tucker format

Tucker format

$$A(i_1, \ldots, i_d) = \sum_{\alpha_1, \ldots, \alpha_d} G(\alpha_1, \ldots, \alpha_d) U_1(i_1, \alpha_1) \ldots U_d(i_d, \alpha_d)$$

- $dnr + r^d$ parameters (high!)
- SVD-based algorithms
- No uniqueness

Main question

Can we find something inbetween? (Tucker and canonical)

The tensor format that has:

- No curse of dimensionality
- SVD-based algorithms

Plan of lecture 2

- History of novel formats
- The Tree-Tucker, Tensor Train, Hierarchical Tucker formats
- Their difference
- Concept of Tensor Networks
- Stability and quasioptimality
- Basic arithmetic (with illustration)
- Cross approximation formula (with illustrations)
- QTT-format (part 1)

History(0)

In 2000-s there was a lot of work done on the canonical/Tucker formats in multilinear algebra:

- Beylkin и Mohlenkamp (2002), first to use as a **format**
- Hackbusch, Khoromskij, Tyrtyshnikov, Grasedyck

History

Beginning of 2009, two papers:

- I. V. Oseledets, E. E. Tyrtyshnikov,

Breaking the curse of dimensionality, or how to use SVD in many dimensions

- W. Hackbusch, S. Kühn, A new scheme for the tensor representation

Two hierarchical schemes:
TT (TT=Tree Tucker) и HT(Hierarchical Tucker)

History

It was almost immediately found, that Tree-Tucker can be rewritten in a much simpler algebraic way, called Tensor-Train.

History

- In March-April 2009 all the basic arithmetics was obtained for the TT-formats, with similar algorithms obtained for HT by different groups later on, but:
  - HT are typically more complex
  - There is no explicit advantage in practice

History

- June 2009 года: L. Grasedyck, Hierarchical singular value decomposition of tensors
- June 2009 года: O., Tyrtyshnikov, TT-cross approximation of multidimensional arrays - first skeleton decomposition formula in many dimensions.

History

- 2010, R. Schneider found that similar things were used in solid state physics (Matrix Product States), as a representation of certain states (but not as a mathematical instruments)
- White (1993), Ostlund и Rommer (1995), Vidal (2003).
- Approaches MCTDH/ML-MCTDH in quantum chemistry can be interperted as a HT-format.
- New mathematical tensor-based framework has emerged

History

The topic is very "hot" and is full of new challenges.

- Merging of linear algebra and many different areas
- Old and new applications
- Numerical experiments are far ahead of the theoretical results
- Limitations?

Tensors and matrices

Idea: if for matrices everything is good, let us transform tensors into matrices!

Tensors and matrices

By reshaping!

$$(i_1, \ldots, i_d) = (\mathcal{I}, \mathcal{J}),$$

$$\mathcal{I} = (i_1, i_4), \quad \mathcal{J} = (i_2, i_3, i_5).$$

$$\mathbf{A} \rightarrow B(\mathcal{I}, \mathcal{J}) \text{ - a matrix}$$

First lemma

Lemma 1

If $\mathbf{A}$ has canonical rank r then for any splitting $B = A(\mathcal{I}, \mathcal{J})$

$$\mathrm{rank}\, B \leq r$$

Second lemma

$B = UV^\top$, still exponentially many parameters!

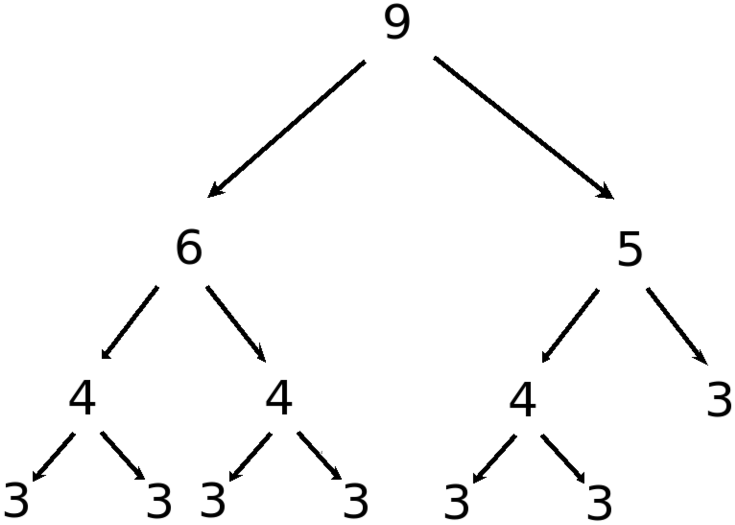
Lemma 2

Let $B = UV^\top$ with full-rank U and V
Then, $U = U(\mathcal{I}, \alpha)$, $V = V(\mathcal{J}, \alpha)$ can be considered as $d_1 + 1$ and $d_2 + 1$ tensors; then these tensors have canonical rank-r representations!

Dimension tree

The process can be then applied recursively: We had a 9 dimensional tensor of canonical rank r, splitted into 4 and 5 indices, then replaced it by $5 = 4 + 1$ and $6 = 5 + 1$ dimensional tensors of canonical rank r. We can go on ...

Dimension tree



Dimension tree

Theorem: The number of leafs (3-d tensors) is exactly $(d-2)$

Complexity is $\mathcal{O}(dnr) + (d-2)r^3$.

Equivalence to the tensor train(1)

We quickly realized, that the tree is in fact **not needed**, and up to the permutation of the dimensions,

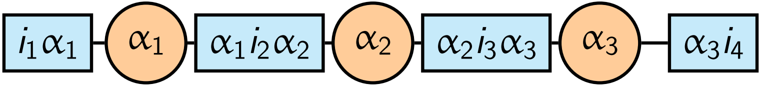
Tensor train

$A(i_1, \dots, i_d) = \sum_{\alpha_1,\dots,\alpha_{d-1}} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \dots G_d(\alpha_{d-1}, i_d)$

Tensor train (2)

Tensor train

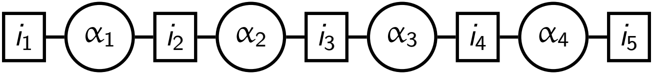
$$A(i_1, \dots, i_d) = \sum_{\alpha_1, \dots, \alpha_{d-1}} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \dots G_d(\alpha_{d-1}, i_d)$$



Tensor train (3)

Tensor train

$$A(i_1, \dots, i_d) = G_1(i_1) G_2(i_2) \dots G_d(i_d).$$



The matrices $G_k(i_k)$ have sizes $r_{k-1} \times r_k$,
$r_0 = r_d = 1$, the numbers r_k are called TT-ranks.

HT format

The Hierachical Tucker format can be treated as sequential application of the Tucker decomposition:

- Compute the Tucker of an $n \times n \times n \times n \times n$ array, get the core $r \times r \times r \times r \times r$
- Select pairs, reshape into a $r^2 \times r^2 \times r^2 \times r$ array
- Compute the Tucker decomposition (again), the factors will be $r_{leaf} r_{leaf} r_{father}$ - the same 3d-tensors
- Do it recursively

The process is described by a binary tree

Tensor network concept

All these formats can be interpreted as tensor networks:

Canonical format

Tucker format

Linear Tensor Network (LTN) - TT-format

Tree Tensor Network - HT/format

What about more complex networks?

Tensor network concept (2)

Multidimensional grids (PEPS-states)

They are not closed!

J. M. Landsburg, Y. Qi, K. Ye, On the geometry of tensor network states, arxiv.org/pdf/1105.4449.pdf

- The multidimensional states can be useful, but we will face all the hazards of the canonical format (again)!

Definition

The tensor is said to be in the TT-format, if

$$A(i_1, \ldots, i_d) = G_1(i_1) G_2(i_2) \ldots G_d(i_d),$$

where $G_k(i_k)$ is a $r_{k-1} \times r_k$ matrix, $r_0 = r_d = 1$

r_k are called TT-ranks

$G_k(i_k)$ (which are in fact $r_{k-1} \times n_k \times r_k$) are called cores

TT in a nutshell

- $\mathbf{A}$ has canonical rank $r \to r_k \leq r$
- TT-ranks are matrix ranks, TT-SVD
- All basic arithmetic, linear in d, polynomial in r
- Fast TENSOR ROUNDING
- TT-cross method, exact interpolation formula
- Q(Quantics, Quantized)-TT decomposition — binarization (or tensorization) of vectors, matrices

TT-ranks are matrix ranks

Define unfoldings:

$A_k = A(i_1 \dots i_k; i_{k+1} \dots i_d)$, $n^k \times n^{d-k}$ matrix

TT-ranks are matrix ranks

Define unfoldings:

$A_k = A(i_1 \dots i_k; i_{k+1} \dots i_d)$, $n^k \times n^{d-k}$ matrix

Theorem: there exists a TT-decomposition with TT-ranks

$$r_k = \operatorname{rank} A_k$$

TT-ranks are matrix ranks

The proof is constructive and gives the TT-SVD algorithm!

TT-ranks are matrix ranks

No exact ranks in practice – stability estimate!

Theorem (Approximation theorem)

If $A_k = R_k + E_k$, $||E_k|| = \varepsilon_k$

$$||\mathbf{A} - \mathbf{TT}||_F \leq \sqrt{\sum_{k=1}^{d-1} \varepsilon_k^2}.$$

TT-SVD

Suppose, we want to approximate:

$$A(i_1, \dots, i_d) \approx G_1(i_1) G_2(i_2) G_3(i_3) G_4(i_4)$$

1. A_1 is an $n_1 \times (n_2 n_3 n_4)$ reshape of $\mathbf{A}$.
2. $U_1, S_1, V_1 = \mathrm{SVD}(A_1)$, U_1 is $n_1 \times r_1$ — first core
3. $A_2 = S_1 V_1^*$, A_2 is $r_1 \times (n_2 n_3 n_4)$.
   Reshape it into a $(r_1 n_2) \times (n_3 n_4)$ matrix
4. Compute its SVD:
   $U_2, S_2, V_2 = \mathrm{SVD}(A_2)$,
   U_2 is $(r_1 n_2) \times r_2$ — second core, V_2 is $r_2 \times (n_3 n_4)$
5. $A_3 = S_2 V_2^*$,
6. Compute its SVD:
   $U_3 S_3 V_3 = \mathrm{SVD}(A_3)$, U_3 is $(r_2 n_3) \times r_3$, V_3 is $r_3 \times n_4$

Fast and trivial linear algebra

Addition, Hadamard product, scalar product, convolution

All scale linear in d

Fast and trivial linear algebra

$$C(i_1, \ldots, i_d) = A(i_1, \ldots, i_d) B(i_1, \ldots, i_d)$$

$$C_k(i_k) = A_k(i_k) \otimes B_k(i_k),$$

ranks are multiplied

Tensor rounding

$\mathbf{A}$ is in the TT-format with suboptimal ranks.
How to reapproximate?

Tensor rounding

ε-rounding can be done in $\mathcal{O}(dnr^3)$ operations

Tensor rounding (detailed)

Everything comes from matrices:

$$A = UV^\top,\ U \in \mathbb{R}^{n \times R}\ V \in \mathbb{R}^{m \times R},$$

Tensor rounding (detailed)

Everything comes from matrices:

$$A = UV^\top,\ U \in \mathbb{R}^{n \times R}\ V \in \mathbb{R}^{m \times R},$$

Rounding

$U = Q_u R_u$, $V = Q_v R_v$
$S = R_u R_v^\top$ (is $R \times R$), $r = \mathrm{rank}\, S$,
$S = \widehat{U} \Lambda \widehat{V}^\top + E, \|E\| \leq \varepsilon$
$A = (Q_u \widehat{U}) \Lambda (Q_v \widehat{V})^\top$ — SVD.
Complexity: $\mathcal{O}((n^k + n^{d-k}) R_k^2 + R_k^3)$.

Tensor rounding (detailed)

Everything comes from matrices:

$$A = UV^\top,\ U \in \mathbb{R}^{n \times R}\ V \in \mathbb{R}^{m \times R},$$

Tensor:

Unfolding $A_k = A(i_1 i_2 \dots i_k; i_{k+1} \dots i_d) = U_k V_k^\top$

$$U_k \in \mathbb{R}^{n^k \times R_k}\ V \in \mathbb{R}^{n^{d-k} \times R_k},$$

QR is not computable in full format

Tensor rounding (detailed)

QR of U_k, V_k can be computed in TT-format in $\mathcal{O}(dnr^3)$ operations!

How it works

How it works

$$U_k(i_1, i_2, \ldots, i_k; \alpha_k) =$$

$$\sum_{\alpha_1,\ldots,\alpha_{k-1}} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \ldots G_k(\alpha_{k-1}, i_k, \alpha_k)$$

First orthogonalize G_1: $G_1(i_1, \alpha_1) = Q_1(i_1, \beta_1) R(\beta_1, \alpha_1)$

How it works

How it works

$$U_k(i_1, i_2, \ldots, i_k; \alpha_k) =$$
$$\sum_{\beta_1, \ldots, \alpha_{k-1}} Q_1(i_1, \beta_1) G_2'(\beta_1, i_2, \alpha_2) \ldots G_k(\alpha_{k-1}, i_k, \alpha_k)$$

Then orthogonalize $G_2'(\beta_1 i_2; \alpha_2)$:

$$G_2'(\beta_1 i_2; \alpha_2) = Q_2(\beta_1, i_2, \beta_2) R(\beta_2, \alpha_2)$$
$$U_k(i_1, i_2, \ldots, i_k; \alpha_k) =$$
$$\sum_{\beta_1, \beta_2 \ldots, \alpha_{k-1}} Q_1(i_1, \beta_1) Q_2(\beta_1, i_2, \beta_2) \ldots G_k(\alpha_{k-1}, i_k, \alpha_k)$$

How it works

How it works

In the end we have

$$U_k(i_1, i_2, \ldots, i_k; \alpha_k) = \sum_{\beta_1,\beta_2\ldots,\beta_{k-1}} Q_1(i_1, \beta_1) Q_2(\beta_1, i_2, \beta_2) \ldots Q_k(\beta_{k-1}, i_k, \beta_k) R(\beta_k, \alpha_k)$$

And that is the QR-decomposition.

Cross approximation in d-dimensions

What if the tensor is given as a "black box"?

Cross approximation in d-dimensions

What if the tensor is given as a "black box"?

O., Tyrtyshnikov, 2010:
TT-cross approximation of multidimensional arrays
You can exactly interpolate rank-r tensor on $\mathcal{O}(dnr^2)$ elements

Making everything a tensor: the QTT

The idea was simple: make everything a tensor (we have software, we have to use it!)

Making everything a tensor: the QTT

Let $f(x)$ be a univariate function (say, $f(x) = \sin x$).

Let v be a vector of values on a uniform grid with 2^d points.

Transform v into a $2 \times 2 \times \ldots \times 2$ d-dimensional tensor.

Compute TT-decomposition of it!

And this is the QTT-format

Putting it all together:

Computing the integral

$$\int_0^\infty \frac{\sin x}{dx} = \frac{\pi}{2}$$

Using the rectangular rule.

Lecture 3

- QTT-format (part 2), application to numerical integration
- QTT-Fourier transform and its relation to tensor networks
- QTT-convolution, explicit representation of Laplace-like tensors
- DMRG/AMEN techniques
- Solution of linear systems in the TT-format
- Solution of eigenvalue problems in the TT-format