Numerical Treatment of Tensors and New Discretisation Paradigms

Wolfgang Hackbusch

Max-Planck-Institut für *Mathematik in den Naturwissenschaften*

Inselstr. 22-26, D-04103 Leipzig, Germany
wh@mis.mpg.de

http://www.mis.mpg.de/scicomp/hackbusch_e.html

Lugano, September 16, 2013

Overview

DDM

— Choice of Subdomains - Discretisation

Tensors - Introduction

— Tensors and Tensor Spaces

— Numerical Tensor Calculus

— Tensor Operations

Tensor Representations

— r-Term Format (Canonical Format)

— Matricisation and α-Ranks

Hierarchical Format

Solution of Linear Systems

Tensorisation

1 DDM

Reasons for Domain Decomposition

- subdomains correspond to a simpler pde (constant coefficients / coefficients of similar size / same pde)

- problems in subdomains easy to solve

- storage distribution

Here:

Assume an elliptic boundary value problem in $\Omega \subset \mathbb{R}^3$ with locally very smooth (analytic) solutions. Point and edge singularities may occur.

Divide Ω into subdomains Ω_ν such that the pde on Ω_ν allows a discretisation by a physically uniform grid; i.e., the grid has nodal points

$$(x_i, y_j, z_k) \in \Omega_\nu \quad \text{for } 1 \le i \le n_1, 1 \le j \le n_2, 1 \le k \le n_3$$

(there are transformations T_ν of a parallelopiped $Q_\nu = [a_1, b_1] \times [a_2, b_2] \times [a_1, b_1]$ onto Ω_ν and $(x_i, y_j, z_k) = T_\nu(\mathbf{x}_0 + (ih_x, jh_y, kh_z))$).

Then one can hope to solve the subproblems with a cost (storage+time) related to

$$\sum_{j=1}^{3} \log n_j = \log(n_1 n_2 n_3).$$

- Better *cost vs. accuracy ratio* than for spectral / p / hp methods.
- Almost no overhead. Black-box approach.
- No standard adaptive approach.
- Direct computations.

Coupling formulation, e.g., by DG:

$$\begin{aligned} a_{DG}(u,v) := &\sum_i \int_{\Omega_i} \langle \nabla u, \nabla v \rangle \, \mathsf{d}x \\ &- \sum_i \int_{\partial\Omega_i} \left\{ \frac{\partial u}{\partial n_i} \right\} [v] - \left\{ \frac{\partial v}{\partial n_i} \right\} [u] \, \mathsf{d}s \\ &+ \eta \sum_i |\partial\Omega_i|^{-1} \int_{\partial\Omega_i} [u] \, [v] \, \mathsf{d}s. \end{aligned}$$

(example for Δ).

This allows an

- inexact solution of the subdomain problems
- inexact evaluation of the boundary data.

The accuracy is less determined by the step size (since it can be rather fine), but by the truncation to certain **tensor ranks**; i.e., the tensor rank is a more relevant parameter than the step size.
There is a direct control of the truncation error by the ranks (via SVD).

2 Tensors - Introduction

2.1 Tensors and Tensor Spaces

Given *vector spaces* V_j for $j = 1, \ldots, d$ (of any dimension), the algebraic tensor space

$$\mathbf{V} := V_1 \otimes V_2 \otimes \ldots \otimes V_d$$

is well-defined. Particular examples of V_j :

function spaces:	$V_j = L^2(\Omega_j)$,
grid functions:	$V_j = \mathbb{R}^{n_j}$,
operators:	$V_j = \mathcal{L}(H_0^1(\Omega_j), H^{-1}(\Omega_j))$, e.g., $\mathbf{\Delta} \in \mathbf{V}$,
matrices:	$V_j = \mathbb{R}^{n_j \times m_j}$.

Any element of a tensor space is called tensor.

Topological tensor space: V_j and $\mathbf{V}$ equipped with certain norms;
$\mathbf{V}$ completed w.r.t. to its norm yields a Banach (or Hilbert) space.

2.2 Numerical Tensor Calculus

1) Representation - Storage:
Original data size of general tensors in $\bigotimes_{j=1}^{d} \mathbb{R}^{n_j}$ is $\prod_{j=1}^{d} n_j$ $(= n^d)$.
Try to represent tensors of interest by data of acceptable size .

Note that the grid functions ('**vectors**') as well as the system **matrices** are considered as tensors.

2) Operations:
Given representations of tensors and an operation (e.g., the matrix-vector multiplication), find the (approximate) representation of the result of the operation.
The standard requirement is that the cost of the algorithm is related to the data sizes of the tensors.

Of particular interest for the present problem, is the solution of linear systems.

2.3 Tensor Operations

addition: $\mathbf{v} + \mathbf{w}$,

scalar product: $\langle \mathbf{v}, \mathbf{w} \rangle$

matrix-vector multiplication: $\left(\bigotimes_{j=1}^{d} A^{(j)} \right) \left(\bigotimes_{j=1}^{d} v^{(j)} \right) = \bigotimes_{j=1}^{d} A^{(j)} v^{(j)}$,

Hadamard product: $(\mathbf{v} \odot \mathbf{w})[\mathbf{i}] = \mathbf{v}[\mathbf{i}]\mathbf{w}[\mathbf{i}]$, pointwise product of functions

$$\left(\bigotimes_{j=1}^{d} v^{(j)} \right) \odot \left(\bigotimes_{j=1}^{d} w^{(j)} \right) = \bigotimes_{j=1}^{d} v^{(j)} \odot w^{(j)},$$

convolution: $\mathbf{v}, \mathbf{w} \in \bigotimes_{j=1}^{d} \mathbb{R}^n : \mathbf{u} = \mathbf{v} \star \mathbf{w}$ with $\mathbf{u_i} = \sum_{\mathbf{0} \leq \mathbf{k} \leq \mathbf{i}} \mathbf{v_{i-k}} \mathbf{w_k}$

$$\left(\bigotimes_{j=1}^{d} v^{(j)} \right) \star \left(\bigotimes_{j=1}^{d} w^{(j)} \right) = \bigotimes_{j=1}^{d} v^{(j)} \star w^{(j)}.$$

3 Tensor Representations

3.1 r-Term Format (Canonical Format)

By definition, each algebraic tensor $\mathbf{v} \in \mathbf{V} = V_1 \otimes V_2 \otimes \ldots \otimes V_d$ has a representation

$$\mathbf{v} = \sum_{\rho=1}^{r} v_\rho^{(1)} \otimes v_\rho^{(2)} \otimes \ldots \otimes v_\rho^{(d)} \qquad \text{with } v_\rho^{(j)} \in V_j$$

and suitable r. Set

$$\mathcal{R}_r := \left\{ \sum_{\rho=1}^{r} v_\rho^{(1)} \otimes v_\rho^{(2)} \otimes \ldots \otimes v_\rho^{(d)} : v_\rho^{(j)} \in V_j \right\}$$

Storage: rdn (for $n = \max \dim V_j$).
If r is of moderate size, this format is advantageous.
Often, a tensor $\mathbf{v}$ is replaced by an approximation $\mathbf{v}_\varepsilon \in \mathcal{R}_r$ with $r = r(\varepsilon)$.

Successful example of an $\mathbf{r}$-term approximation

The discrete Laplace operator (any separable operator) is of the form

$$\mathbf{A} = T_1 \otimes I \otimes \ldots \otimes I + \ldots + I \otimes \ldots \otimes I \otimes T_d.$$

Solution of discrete Poisson problem: $\mathbf{A}^{-1} \approx \mathbf{B}_r$ with $\mathbf{B}_r$ of the form

$$\mathbf{B}_r = \sum_{i=1}^{r} a_i \bigotimes_{j=1}^{d} \exp(-b_i T_j) \in \mathcal{R}_r\,,$$

where $a_i, b_i > 0$ are explicitly known.
Helpful for preconditioning.

Error estimate:

$$\left\| \mathbf{B}_r - \mathbf{A}^{-1} \right\|_2 \leq O(\exp(-cr^{1/2})).$$

Easy to compute, even for $T_j \in \mathbb{R}^{1000\times1000}$ and $d = 1000$; i.e., $\mathbf{B}_r \in \mathbb{R}^{M\times M}$ with $M = 1000^{1000}$.

3.2 Matricisation and α-Ranks

$V_j = \mathbb{R}^{n_j}$, $\mathbf{V} = \bigotimes_{j=1}^{d} \mathbb{R}^{n_j}$, $D := \{1, \ldots, d\}$.

Given a true subset $\alpha \subset D$, regroup the indices of $\mathbf{v}[i_1, \ldots, i_d]$ into the two tuples $\mathbf{i}_\alpha := (i_j : j \in \alpha)$ and $\mathbf{i}_{\alpha^c}$, where $\alpha^c := D \backslash \alpha$.

Matricisation:

$$\mathbf{v} \mapsto M_\alpha := \mathcal{M}_\alpha(\mathbf{v}) \in \mathbb{R}^{n_\alpha \times n_{\alpha^c}}$$

with $n_\beta := \prod_{j \in \beta} n_j$ defined by the entries

$$M_\alpha[\mathbf{i}_\alpha, \mathbf{i}_{\alpha^c}] := \mathbf{v}[i_1, \ldots, i_d].$$

α-th rank:

$$\mathrm{rank}_\alpha(\mathbf{v}) := \mathrm{rank}(\mathcal{M}_\alpha(\mathbf{v})).$$

4 Hierarchical Format

4.1 Dimension Partition Tree

Example: $\mathbf{v} \in \mathbf{V} = V_1 \otimes V_2 \otimes V_3 \otimes V_4$. There are subspaces

$$U_1 \subset V_1, \qquad U_2 \subset V_2, \qquad U_3 \subset V_3, \qquad U_4 \subset V_4,$$

$$\mathbf{U}_{\{1,2\}} \subset V_1 \otimes V_2, \qquad \mathbf{U}_{\{3,4\}} \subset V_3 \otimes V_4$$

such that

$$\mathbf{v} \in \mathrm{span}\{\mathbf{v}\} \subset \mathbf{U}_{\{1,2\}} \otimes \mathbf{U}_{\{3,4\}} \subset \mathbf{V}$$

$$\mathbf{U}_{\{1,2\}} \subset U_1 \otimes U_2 \qquad \mathbf{U}_{\{3,4\}} \subset U_3 \otimes U_4$$

$$U_1 \subset V_1 \qquad U_2 \subset V_2 \qquad U_3 \subset V_3 \qquad U_4 \subset V_4$$

There are optimal subspaces $\mathbf{U}_\alpha := U_\alpha^{\mathsf{min}}(\mathbf{v})$ with $\dim \mathbf{U}_\alpha = \mathrm{rank}_\alpha(\mathbf{v})$.

Dimension partition tree:
Any binary tree T_D with root $D := \{1, \ldots, d\}$ and leaves $\{1\}, \{2\}, \ldots, \{d\}$.

4.2 Algorithmic Realisation

Typical situation: $\mathbf{U}_{\{1,2\}} \subset U_1 \otimes U_2$ (nestedness property).

Bases: $U_1 = \operatorname*{span}_{1\le i\le r_1} \{b_i^{(1)}\}$, $U_2 = \operatorname*{span}_{1\le j\le r_2} \{b_j^{(2)}\}$, $\mathbf{U}_{\{1,2\}} = \operatorname*{span}_{1\le \ell\le r_{\{1,2\}}} \{\mathbf{b}_\ell^{(\{1,2\})}\}$.

$$\mathbf{b}_\ell^{(\{1,2\})} = \sum_{i=1}^{r_1}\sum_{j=1}^{r_2} c_{ij}^{(\{1,2\},\ell)}\, b_i^{(1)} \otimes b_j^{(2)}$$

Basis vectors $b_\nu^{(j)} \in V_j$ $(1 \le j \le d)$ are explicitly stored, for other nodes store the coefficient matrices

$$C^{(\alpha,\ell)} = \left(c_{ij}^{(\alpha,\ell)}\right)_{ij} \in \mathbb{K}^{r_{\alpha_1}\times r_{\alpha_2}}.$$

The tensor is represented by $\mathbf{v} = c_1 \mathbf{b}_1^{(D)}$.

$\mathfrak{r} = (r_\alpha)_{\alpha\in T_D}$ tuple of ranks. Then

$$\mathcal{H}_\mathfrak{r} = \{\mathbf{v} \in \mathbf{V} : \operatorname{rank}_\alpha(\mathbf{v}) \le r_\alpha \text{ for all } \alpha \in T_D\} \tag{1}$$

Properties of $\mathcal{H}_{\mathfrak{r}}$

1) *Storage*: $r := \max_\alpha r_\alpha$; $n := \max_j \dim(V_j)$,

$$(d-1)r^3 + drn$$

Linearity in d.

2) *Operations*: recursive algorithm, typical operation cost:

$$O(dr^4 + dnr^2)$$

Furthermore:

- HOSVD (higher-order singular value decomposition),
- quasi-optimal truncation,
- numerical stability

Case of $d = 3$

```
                 {1,2,3}
                /       \
Tree:      {1,2}         3
           /   \
          1     2
```

Trace of $v(x, y, z)$ at $z = z_0$:

1) evaluate the basis vectors $\{b_j^{(3)} : 1 \leq j \leq r_3\}$ of U_3 at $z = z_0$;

2) $\mathbf{v}|_{z=z_0} = c_1 \mathbf{b}_1^{(D)}|_{z=z_0} = c_1 \sum_{i=1}^{r_{\{1,2\}}} \left(\sum_{j=1}^{r_3} c_{ij}^{(D,1)} \, b_j^{(3)}|_{z=z_0} \right) b_i^{(\{1,2\})} \in U_{\{1,2\}}$.

Similar for traces at $x = x_0$, $y = y_0$ or for Neumann boundary values.

5 Solution of Linear Systems

Linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b},$$

where $\mathbf{x}, \mathbf{b} \in \mathbf{V} = \bigotimes_{j=1}^{d} V_j$ and $\mathbf{A} \in \bigotimes_{j=1}^{d} \mathcal{L}(V_j, V_j) \subset \mathcal{L}(\mathbf{V}, \mathbf{V})$ are represented in one of the formats (e.g., $\mathbf{A}$: r-term format, $\mathbf{x}, \mathbf{b}$: hierarchical format):

Standard linear iteration:

$$\mathbf{x}^{m+1} = \mathbf{x}^{m} - \mathbf{B}\left(\mathbf{A}\mathbf{x} - \mathbf{b}\right).$$

$\Rightarrow$ representation ranks blow up.

Therefore truncations T are used ('truncated iteration'):

$$\mathbf{x}^{m+1} = T\left(\mathbf{x}^{m} - \mathbf{B}\left(T\left(\mathbf{A}\mathbf{x} - \mathbf{b}\right)\right)\right).$$

Cost per step: $nd \times$ powers of the involved representation ranks.

$$\mathbf{x}^{m+1} = T\left(\mathbf{x}^m - \mathbf{B}\left(T\left(\mathbf{A}\mathbf{x} - \mathbf{b}\right)\right)\right)$$

Choice of $\mathbf{B}$**:**

If $\mathbf{A}$ corresponds to an elliptic pde of order 2, choose a *separable, spectrally equivalent* $\tilde{\mathbf{A}} \Rightarrow \mathbf{B} \approx \tilde{\mathbf{A}}^{-1}$ has a simple r-term format.

Obvious variants: cg-like methods

Literature:

Khoromskij 2009, Kressner-Tobler 2010, Kressner-Tobler 2011 (SIAM), Kressner-Tobler 2011 (CMAM), Osedelets-Tyrtyshnikov-Zamarashkin 2011, Ballani-Grasedyck 2013, Savas-Eldén 2013

Remark: For $d = 2$, these linear systems may be written as matrix equations:

$$(A \otimes I + I \otimes A)\,\mathbf{x} = \mathbf{b} \quad \Leftrightarrow \quad AX + XA = B \qquad \text{(Lyapunov)}$$

(cf. Benner-Breiten 2013).

Variational Approach

Define

$$\Phi(\mathbf{x}) := \langle \mathbf{Ax}, \mathbf{x} \rangle - 2 \langle \mathbf{b}, \mathbf{x} \rangle$$

if $\mathbf{A}$ is positive definite or

$$\Phi(\mathbf{x}) := \|\mathbf{Ax} - \mathbf{b}\|^2 \qquad \text{or}$$
$$\Phi(\mathbf{x}) := \|\mathbf{B}(\mathbf{Ax} - \mathbf{b})\|^2$$

and try to minimise $\Phi(\mathbf{x})$ over all parameters of $\mathbf{x}$ appearing in a fixed format.

Literature:
Espig-Hackbusch-Rohwedder-Schneider, Falcó-Nouy,
Holtz-Rohwedder-Schneider, Mohlenkamp, Osedelets,...

So far:

$$\begin{aligned} cost &= O(\text{number of iterations} \cdot d \cdot \left[rank^4 + n \cdot rank^2\right]), \\ n &: \text{number of nodal points in } \textit{one} \text{ direction.} \end{aligned}$$

6 Tensorisation

$V_j = \mathbb{R}^n \Rightarrow$ storage: $rdn + (d-1)r^3$. Now: $n \to O(\log n)$

Let the vector $y \in \mathbb{R}^n$ represent the grid values of a function in $(0, 1]$:

$$y_\mu = f\left(\frac{\mu+1}{n}\right) \qquad (0 \le \mu \le n-1)\,.$$

Choose, e.g., $n = 2^d$, and note that $\mathbb{R}^n \cong \mathbf{V} := \bigotimes_{j=1}^d \mathbb{R}^2$.
Isomorphism by binary integer representation:
$\mu = \sum_{j=1}^d \mu_j 2^{j-1}$ with $\mu_j \in \{0, 1\}$, i.e.,

$$y_\mu = \mathbf{v}[\mu_1, \mu_2, \ldots, \mu_{d-1}, \mu_d].$$

Algebraic Function Compression (black-box procedure)

1) Tensorisation: $y \in \mathbb{R}^n \longmapsto \mathbf{v} \in \mathbf{V}$ (storage size: $n = 2^d$)

2) Apply the *tensor truncation*: $\mathbf{v} \longmapsto \mathbf{v}_\varepsilon$

3) Observation: often the data size decreases from $n = 2^d$ to $O(d) = O(\log n)$.

EXAMPLE

$y \in \mathbb{C}^n$ with $y_\mu = \zeta^\mu$ leads to an *elementary tensor* $\mathbf{v} \in \mathbf{V}$, i.e.,

$$\mathbf{v} = \bigotimes_{j=1}^{d} v^{(j)} \qquad \text{with } v^{(j)} = \begin{bmatrix} 1 \\ \zeta^{2^{j-1}} \end{bmatrix} \in \mathbb{C}^2.$$

Storage size $= 2d = 2\log_2 n$.

Consequence:

All functions $f \in C((0,1])$, which can be well-approximated by r *trigonometric terms* or *exponential sums* with r terms (even with complex coefficients) can be approximated by a tensor representation with data size

$$2dr = O(r \log n).$$

Hierarchical Format, Matricisation

Tree:

$$\begin{array}{ccccccccccc}
1 & — & \{1,2\} & — & \{1,2,3\} & — & \dots & \{1,\dots,d\text{-}1\} & — & \{1,\dots,d\} \\
 & / & & / & & / & \dots & & / & \\
2 & & 3 & & 4 & & \dots & d & &
\end{array}$$

(also called TT format)

Consider the tensorisation $\mathbf{v} \in \bigotimes_{j=1}^{d} \mathbb{R}^2$ of the vector $y = (y_0, \dots, y_{n-1}) \in \mathbb{R}^n$.
The matricisation for $\alpha = \{1, \dots, j\}$ $(1 \leq j \leq d-1)$ yields

$$\mathcal{M}_\alpha(\mathbf{v}) = \begin{bmatrix} y_0 & y_m & \cdots & y_{n-m} \\ y_1 & y_{m+1} & \cdots & y_{n-m+1} \\ \vdots & \vdots & & \vdots \\ y_{m-1} & y_{2m-1} & \cdots & y_{n-1} \end{bmatrix} \text{ with } m := 2^j.$$

Recall: $\mathrm{rank}_\alpha(\mathbf{v}) = \dim \mathcal{M}_\alpha(\mathbf{v})$.

Polynomials

f polynomial of degree $p \Rightarrow \text{rank}_\alpha(\mathbf{v}) = \dim \mathcal{M}_\alpha(\mathbf{v}) \leq p + 1$.

hp method, i.e., piecewise polynomial

Singularity at $x = 0$, partition:

$$[0, \tfrac{1}{n}],\ [\tfrac{1}{n}, \tfrac{2}{n}],\ [\tfrac{2}{n}, \tfrac{4}{n}], \ldots, [\tfrac{1}{4}, \tfrac{1}{2}],\ [\tfrac{1}{2}, 1].$$

Local polynomials of degree $p \Rightarrow \text{rank}_\alpha(\mathbf{v}) = \dim \mathcal{M}_\alpha(\mathbf{v}) \leq p + 2$.

Conclusion: If any hp approximation with a piecewise polynomial $\mathbf{P}$ of degree $\leq p$ exists, then the tensorised grid function $\mathbf{f}$ can be approximated by a tensor $\tilde{\mathbf{f}}$ such that the ranks are bounded by $p + 2$ and

$$\left\| \mathbf{f} - \tilde{\mathbf{f}} \right\|_2 \leq \| \mathbf{f} - \mathbf{P} \|_2$$

The data size is bounded by

$$\leq 2d\,(p + 2)^2\,.$$

The computation of $\tilde{\mathbf{f}}$ is completely black-box (e.g., no information about location of the singularity required).

Treatment of Multivariate (Grid) Functions

So far, basis vectors

$$\{b_\nu^{(j)} : 1 \leq \nu \leq r_j\} \subset U_j$$

are required for all directions $1 \leq j \leq d (= 3)$.

Assume $n_j = 2^{d_j}$ for the size of $b_\nu^{(j)} \in \mathbb{R}^{n_j}$ and use the tensorised representation.

The data size may decrease from $O(\left[r^3 + n \cdot r\right])$ to $O(r^3 + r \log n)$.
The corresponding cost of operations is

$$O(r^4 + r^2 \log n).$$

This allows the use of very large n; e.g., $n = 2^{30} = 1,073,741,824$.

Consequence: **no adaptive discretisation**.

Conclusions

Standard Paradigm:

- Work more or less proportional to the dimension of the ansatz space
- Given an accuracy requirement, minimise this dimension
- consequence: adaptive approaches, hp ansatz, big overhead

Instead for the case of Cartesian domain:

- Uniform discretisation with very fine step size
- Tensorisation leads to reduction to logarithmic work
- almost no overhead / no adaptivity
- basic operations: standard matrix operations, QR, SVD (size: rank)
- error control mainly by tensor truncations to suitable tensor ranks.

DDM: create subdomains which are images of Cartesian domains.

7 Literature

W. Hackbusch: Tensor spaces and numerical tensor calculus. Springer Berlin, 2012