ZERO ANNOUNCEMENT

3rd International Conference on Structured Matrices and Tensors Hong Kong, HKBU and PolyU 19-22 January 2010

CONTACT E-MAIL: mng@math.hkbu.edu.hk (Michael Ng)

SUBLINEAR ALGEBRA FOR TENSOR APPROXIMATIONS OF VECTORS AND MATRICES

Eugene Tyrtyshnikov

tee@inm.ras.ru

Institute of Numerical Mathematics
Russian Academy of Sciences

(joint work with I. Oseledets and D. Savostyanov)

OVERVIEW OF THE TALK

- Huge-scale data calls for *sublinear* complexity
- Tensor background
  - Tucker decomposition
  - Canonical decomposition
- General rank reduction methods
  - Tensor SVD (Higher Order SVD)
  - Alternating Least Squares (ALS)
- Recompression methods in matrix-by-matrix multiplication
  - Pre-recompression stage
  - Tucker-Tucker recompression algorithms
- Matrix inversion with sublinear complexity
  - Newton–Schultz and its modification
  - General theory for approximate iterations
  - Examples: 3D Laplacian and Newton potential
  - Inversion of a two-level Toeplitz matrix
  - Tensor eigensolver perspectives
- Concluding remarks and future work

WHAT IS A HUGE-SCALE PROBLEM?

Solve

$$\int_D \frac{1}{|x-y|}\varphi(y)dy = f(x), \qquad x, y \in D = [0,1]^d$$

Let $d = 3$. Subdive the cube D into subcubes D_{ijk}:

$$D_{ijk} = [a_{i-1}, a_i] \times [a_{j-1}, a_j] \times [a_{k-1}, a_k], \quad 0 = a_0 < a_1 < \ldots < a_n = 1$$

$$\varphi(y) \approx u_{ijk} = \mathbf{const} \quad \text{на} \quad D_{ijk} \quad \text{(collocation)} \quad \Rightarrow \quad Au = f$$

Vectors = discrete functions u_{ijk}, f_{ijk} on a grid with n^3 nodes.
Matrix A contains n^6 nonzero entries, none of them can be neglected, no visible structure in the case of nonuniform grids.

$n = 64 \quad \Rightarrow$ STORAGE FOR A = **512Gb** – not few!

$n = 256 \quad \Rightarrow$ STORAGE FOR A = **2**Pb ($\mathbf{1}$ Pb = $\mathbf{2}^{50}$ byte).

A big problem is already with storage for matrix coefficients!

CAN WE STILL SOLVE IT?

The only idea is to find a sufficiently close problem
with a flaunting low-parametric structure.

Store only few parameters of this structure.

Apply gain-of-the-structure methods.

USE SMOOTHNESS IN DATA!

SMOOTHNESS = RANK STRUCTURES =
TENSOR STRUCTURES

WHAT IS A TENSOR PROBLEM?

It is one where all data on input and output are given exactly or approximately in tensor formats defined by a small number of parameters compared to the total amount of data.

For such problems we propose to seek for algorithms that work with data exclusively in tensor formats, the price we pay is a contamination of data through recompression (approximation) at each operation.

WHAT ARE TENSORS?

TENSOR = MULTI-INDEX ARRAY = MULTI-WAY ARRAY = MULTI-DIMENSIONAL MATRIX:

$$\boldsymbol{A = [a_{ij\ldots k}]}$$

$$\boldsymbol{i \in I, \quad j \in J, \quad \ldots, \quad k \in K}$$

Number of different indices is *dimension*.

Indices are called also *modes*.

Cardinalities of index ranges $\boldsymbol{I, J, \ldots, K}$ are *mode sizes*.

In case of dimension $\boldsymbol{d}$ and mode sizes $\boldsymbol{n_1, n_2, \ldots, n_d}$,
$\boldsymbol{A}$ is a *tensor of size* $\boldsymbol{n_1 \times n_2 \times \ldots \times n_d}$.

Talking of tensors, tacitly assume that $\boldsymbol{d \geq 3}$.

TENSORS AND MATRICES

Let $\boldsymbol{A} = [\boldsymbol{a}_{ijklm}]$.
Consider pairs of complementary *long indices*

$$(\boldsymbol{ij}) \quad \text{and} \quad (\boldsymbol{klm})$$

$$(\boldsymbol{kl}) \quad \text{and} \quad (\boldsymbol{ijm})$$

$$\cdots\cdots\cdots$$

Then $\boldsymbol{A}$ gives rise to several matrices:

$$\boldsymbol{B}_1 = [\boldsymbol{b}_{(ij),(klm)}],$$

$$\boldsymbol{B}_2 = [\boldsymbol{b}_{(kl),(ijm)}]$$

$$\cdots\cdots\cdots$$

with

$$\boldsymbol{b}_{(ij),(klm)} = \boldsymbol{b}_{(kl),(ijm)} = \ldots = \boldsymbol{a}_{ijklm}$$

MODE UNFOLDING MATRICES

$$\boldsymbol{A}_1 = [\boldsymbol{a}_{i,(jklm)}]$$
$$\boldsymbol{A}_2 = [\boldsymbol{a}_{j,(iklm)}]$$
$$\boldsymbol{A}_3 = [\boldsymbol{a}_{k,(ijlm)}]$$
$$\boldsymbol{A}_4 = [\boldsymbol{a}_{l,(ijkm)}]$$
$$\boldsymbol{A}_5 = [\boldsymbol{a}_{m,(ijkl)}]$$

Columns of unfolding matrices are called *mode vectors*.

If $\boldsymbol{d} = \boldsymbol{3}$, typical names are *columns, rows, fibers*.

Ranks of unfolding matrices are called *mode ranks* or *Tucker ranks*.

L. R. Tucker, Some mathematical notes on three-mode factor analysis, *Psychometrika*, V. 31, P. 279–311 (1966).

TENSOR-BY-MATRIX MULTIPLICATIONS

Also called *mode contractions*.

Given a tensor $\boldsymbol{A} = [\boldsymbol{a}_{ijk}]$ and matrices

$$\boldsymbol{U} = [\boldsymbol{u}_{i'i}], \quad \boldsymbol{V} = [\boldsymbol{v}_{j'j}], \quad \boldsymbol{W} = [\boldsymbol{w}_{k'k}],$$

define new tensors

$$\boldsymbol{A}^{U} = \boldsymbol{A} \times_1 \boldsymbol{U} = [\boldsymbol{a}^{U}_{i'jk}]$$
$$\boldsymbol{A}^{V} = \boldsymbol{A} \times_2 \boldsymbol{V} = [\boldsymbol{a}^{V}_{ij'k}]$$
$$\boldsymbol{A}^{W} = \boldsymbol{A} \times_3 \boldsymbol{W} = [\boldsymbol{a}^{W}_{ijk'}]$$

as follows:

$$\boldsymbol{a}^{U}_{i'jk} = \sum_{i} \boldsymbol{u}_{i'i}\, \boldsymbol{a}_{ijk} \quad \Leftrightarrow \quad \boldsymbol{A}^{U}_{1} = \boldsymbol{U}\boldsymbol{A}_1$$
$$\boldsymbol{a}^{V}_{ij'k} = \sum_{j} \boldsymbol{v}_{j'j}\, \boldsymbol{a}_{ijk} \quad \Leftrightarrow \quad \boldsymbol{A}^{V}_{2} = \boldsymbol{V}\boldsymbol{A}_2$$
$$\boldsymbol{a}^{W}_{ijk'} = \sum_{k} \boldsymbol{w}_{k'k}\, \boldsymbol{a}_{ijk} \quad \Leftrightarrow \quad \boldsymbol{A}^{W}_{3} = \boldsymbol{W}\boldsymbol{A}_3$$

WHY CONTRACTIONS?

Let $\boldsymbol{A} = [\boldsymbol{a}_{ijk}]$ be $\boldsymbol{n} \times \boldsymbol{n} \times \boldsymbol{n}$ and mode ranks be equal to $\boldsymbol{r} \ll \boldsymbol{n}$.
Consider $\boldsymbol{QR}$ decompositions of unfolding matrices

$$\boldsymbol{A}_1 = \boldsymbol{Q}_1 \boldsymbol{R}_1, \quad \boldsymbol{A}_2 = \boldsymbol{Q}_2 \boldsymbol{R}_2, \quad \boldsymbol{A}_3 = \boldsymbol{Q}_3 \boldsymbol{R}_2$$

$\boldsymbol{Q}_1, \boldsymbol{Q}_2, \boldsymbol{Q}_3$ are orthogonal $\boldsymbol{n} \times \boldsymbol{r}$ matrices.

Define the *Tucker core* tensor $\boldsymbol{G} = [\boldsymbol{g}_{\alpha\beta\gamma}]$
of *contracted* size $\boldsymbol{r} \times \boldsymbol{r} \times \boldsymbol{r}$:

$$\boldsymbol{G} = \boldsymbol{A} \times_1 \boldsymbol{Q}_1^\top \times_2 \boldsymbol{Q}_2^\top \times_3 \boldsymbol{Q}_3^\top \quad \text{i.e.} \quad \boldsymbol{g}_{\alpha\beta\gamma} = \sum_{i,j,k} \boldsymbol{a}_{ijk}\, \boldsymbol{q}^1_{i\alpha}\, \boldsymbol{q}^2_{j\beta}\, \boldsymbol{q}^3_{k\gamma}$$

THEOREM

$$\boldsymbol{A} = \boldsymbol{G} \times_1 \boldsymbol{Q}_1 \times_2 \boldsymbol{Q}_2 \times_3 \boldsymbol{Q}_3 \quad \text{i.e.} \quad \boldsymbol{a}_{ijk} = \sum_{\alpha,\beta,\gamma} \boldsymbol{g}_{\alpha\beta\gamma}\, \boldsymbol{q}^1_{i\alpha}\, \boldsymbol{q}^2_{j\beta}\, \boldsymbol{q}^3_{k\gamma}$$

IMPORTANT: $\boldsymbol{A}$ is now represented in a *contracted form*
with only $\boldsymbol{3nr + r^3 \ll n^3}$ parameters.

TUCKER DECOMPOSITION

Regarded as *Tensor SVD* or *Higher Order SVD*:

$$\boldsymbol{A} \;=\; \boldsymbol{G} \times_1 \boldsymbol{Q}_1 \times_2 \boldsymbol{Q}_2 \times_3 \boldsymbol{Q}_3 \quad \text{i.e.} \quad \boldsymbol{a}_{ijk} \;=\; \sum_{\alpha,\beta,\gamma} \boldsymbol{g}_{\alpha\beta\gamma}\, \boldsymbol{q}^1_{i\alpha}\, \boldsymbol{q}^2_{j\beta}\, \boldsymbol{q}^3_{k\gamma}$$

Orthogonal matrices $\boldsymbol{Q}_1, \boldsymbol{Q}_2, \boldsymbol{Q}_3$ are *Tucker factors* or *frame matrices*.

THEOREM

Rows in each of unfolding matrices for the Tucker core can be made *orthogonal* and arranged in *length-decreasing order*.
Row lengths of unfoldings for $\boldsymbol{G}$ = singular values of unfoldings for $\boldsymbol{A}$.

PROOF is easy via SVD of unfolding matrices:

if $\boldsymbol{A}_1 = \boldsymbol{Q}_1 \boldsymbol{\Sigma}_1 \boldsymbol{V}_1$ then $(\boldsymbol{A} \times_1 \boldsymbol{Q}_1^\top)_1 = \boldsymbol{\Sigma}_1 \boldsymbol{V}_1.$

Same for other modes.

TUCKER APPROXIMATIONS

$$\boldsymbol{a_{ijk} \approx \sum_{\alpha,\beta,\gamma} g_{\alpha\beta\gamma}\, q^1_{i\alpha}\, q^2_{j\beta}\, q^3_{k\gamma}}$$

APPLICATIONS:

- Multi-way Principal Component Analysis
  (senior frame matrices are most informative).
- Tensor data compression
  (ignore small and get to reduced Tucker ranks $\ll$ mode sizes).
- New generation of numerical algorithms
  with all data in the Tucker format.
  Enjoy linear and even sublinear complexity in total size of data
  (could be petabytes).

I. Oseledets, D. Savostyanov, E. Tyrtyshnikov,
Linear algbera for tensor problems, submitted to *Computing* (2008).

G. Beylkin, M. Mohlenkamp, Algorithms for numerical analysis in high dimensions, *SIAM J. Sci. Comput.*, 26 (6), pp. 2133-2159 (2005).

CANONICAL DECOMPOSITION

$$\boldsymbol{a_{ij\ldots k} = \sum_{t=1}^{\rho} u_{it}\, v_{jt}\ldots\, w_{kt}}$$

Minimal $\boldsymbol{\rho} = \mathbf{tRank}$ is called *canonical rank* or *tensor rank* of $\boldsymbol{A}$.

THEOREM

Let mode ranks be egual to $\boldsymbol{r}$. Then

$$\boldsymbol{r \leq \mathrm{tRank}(A) \leq r^2.}$$

CANONICAL APPROXIMATIONS

$$\boldsymbol{a_{ij\ldots k} \approx \sum_{t=1}^{\rho} u_{it}\, v_{jt}\ldots\, w_{kt}}$$

play same compression role as Tucker.

Could be better but not necessarily!

TENSOR RANKS IN COMPLEXITY THEORY

In the "row-by-column" rule for multiplication of $n \times n$ matrices we have n^2 multiplications. Can we reduce this number?

$$\begin{bmatrix} c_1 & c_2 \\ c_3 & c_4 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} = \begin{bmatrix} b_1 & b_2 \\ b_3 & b_4 \end{bmatrix}$$

$$c_k = \sum_{i=1}^{n} \sum_{j=1}^{n} h_{ijk}\, a_i\, b_j$$

Let $\rho =$ tensor rank of h_{ijk} and canonical decomposition read

$$h_{ijk} = \sum_{t=1}^{\rho} u_{it}\, v_{jt}\, w_{kt} \quad \Rightarrow$$

$$c_k = \sum_{t=1}^{\rho} w_{kt} \left(\sum_{i=1}^{4} u_{it} a_i \right) \left(\sum_{j=1}^{n} v_{jt} b_j \right)$$

Now we have ρ multiplications!

If $n = 2$ then $\rho = 7$ (Strassen, 1965).
By recursion $\Rightarrow$ only $O(n^{\log_2 7})$ multiplications for arbitrary n.

TUCKER VS CANONICAL FOR MATRICES

$$\boldsymbol{a_{ij}} = \sum_{\alpha=1}^{r} \sum_{\beta=1}^{r} \boldsymbol{g_{\alpha\beta} q^1_{i\alpha} q^2_{j\beta}} \quad \Leftrightarrow \quad \boldsymbol{A} = \boldsymbol{Q_1 G Q_2^\top}$$

Tucker = a *pseudo-skeleton* decomposition of $\boldsymbol{A}$.

$$\boldsymbol{a_{ij}} = \sum_{t=1}^{\rho} \boldsymbol{u_{it} v_{jt}} \quad \Leftrightarrow \quad \boldsymbol{A} = \boldsymbol{U V^\top}$$

Canonical = a *skeleton* or *dyadic* decomposition of $\boldsymbol{A}$.

Tensor (canonical) rank seems to be a true generalization of the matrix rank concept.

However, tensor rank for dimension ≥ 3 and matrix rank have *noticably different properties*.

KRONECKER PRODUCT REPRESENTATION

Tucker decomposition:

$$\boldsymbol{A} = \sum_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \boldsymbol{g}_{\alpha\beta\gamma}\, \boldsymbol{u}_{\alpha} \otimes \boldsymbol{v}_{\beta} \otimes \boldsymbol{w}_{\gamma}$$

Canonical decomposition:

$$\boldsymbol{A} = \sum_{\boldsymbol{t}} \boldsymbol{u}_{t} \otimes \boldsymbol{v}_{t} \otimes \boldsymbol{w}_{t}$$

BASIC OPERATION

Given two matrices $\boldsymbol{A}$ *and* $\boldsymbol{X}$*, compute a tensor-structured approximation* $\widetilde{\boldsymbol{Y}}$ *to their product*

$$\widetilde{\boldsymbol{Y}} \approx \boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X}$$

so that $\boldsymbol{Y}$ *never appears as a full matrix.*

Assume that $\boldsymbol{A}$ is $\boldsymbol{N} \times \boldsymbol{N}$.

Important cases for $\boldsymbol{X}$:

(a) $\boldsymbol{X}$ is a vector (a rectangular matrix of size $\boldsymbol{N} \times \boldsymbol{1}$);

(b) $\boldsymbol{X}$ is a square matrix of the same order $\boldsymbol{N}$.

Case (a) is a basic operation in all iterative solvers
for linear systems or eigenvalue problems with the coefficient matrix $\boldsymbol{A}$.

Case (b) is a workhorse operation in computation of matrix functions of $\boldsymbol{A}$,
in particular $\boldsymbol{A}^{-\boldsymbol{1}}$.

CANONICAL FORMAT

$$\boldsymbol{A} = \sum_{t=1}^{\rho} \boldsymbol{A}_t \otimes \boldsymbol{B}_t \otimes \boldsymbol{C}_t$$

$$\boldsymbol{X} = \sum_{\tau=1}^{\rho} \boldsymbol{U}_\tau \otimes \boldsymbol{V}_\tau \otimes \boldsymbol{W}_\tau$$

TUCKER FORMAT

$$\boldsymbol{A} = \sum_{\sigma=1}^{r_1} \sum_{\delta=1}^{r_2} \sum_{\tau=1}^{r_3} f_{\sigma\delta\tau} \boldsymbol{A}_\sigma \otimes \boldsymbol{B}_\delta \otimes \boldsymbol{C}_\tau$$

$$\boldsymbol{X} = \sum_{\alpha=1}^{r_1} \sum_{\beta=1}^{r_2} \sum_{\gamma=1}^{r_3} g_{\alpha\beta\gamma} \boldsymbol{U}_\alpha \otimes \boldsymbol{V}_\beta \otimes \boldsymbol{W}_\gamma$$

Thus, $\boldsymbol{N} = \boldsymbol{n}_1 \boldsymbol{n}_2 \ldots \boldsymbol{n}_d$.

Proceed as if $\boldsymbol{d} = \mathbf{3}$, $\quad \boldsymbol{n} = \boldsymbol{n}_1 = \ldots = \boldsymbol{n}_d$, $\quad \boldsymbol{r} = \boldsymbol{r}_1 = \ldots = \boldsymbol{r}_d$.

COMBINATIONS OF FORMATS

(CC) $\boldsymbol{A}$ and $\boldsymbol{X}$ are both in the canonical.

(CT) $\boldsymbol{A}$ is in the canonical, $\boldsymbol{X}$ is in the Tucker.

(TT) $\boldsymbol{A}$ and $\boldsymbol{X}$ are both in the Tucker.

The result $\widetilde{\boldsymbol{Y}} \approx \boldsymbol{Y} = \boldsymbol{AX}$ is assumed to keep the format of $\boldsymbol{X}$.

We focuse only on $\boldsymbol{CT}$ and $\boldsymbol{TT}$.

PRE-RECOMPRESSION STAGE

$$\boldsymbol{A} = \sum_{\sigma,\delta,\tau} \boldsymbol{f}_{\sigma\delta\tau}\, \boldsymbol{A}_{\sigma} \otimes\ \boldsymbol{B}_{\delta} \otimes\ \boldsymbol{C}_{\tau}$$

$$\boldsymbol{X} = \sum_{\alpha,\beta,\gamma} \boldsymbol{g}_{\alpha\beta\gamma}\, \boldsymbol{U}_{\alpha} \otimes\ \boldsymbol{V}_{\beta} \otimes\ \boldsymbol{W}_{\gamma}$$

$$\Rightarrow \quad \boldsymbol{Y} = \sum_{\sigma,\delta,\tau} \sum_{\alpha,\beta,\gamma} \boldsymbol{f}_{\sigma\delta\tau}\, \boldsymbol{g}_{\alpha\beta\gamma}\, \boldsymbol{A}_{\sigma}\boldsymbol{U}_{\alpha} \otimes\ \boldsymbol{B}_{\delta}\boldsymbol{V}_{\beta} \otimes\ \boldsymbol{C}_{\tau}\boldsymbol{W}_{\gamma}$$

Pre-recompression stage consists in the computation of matrices

$$\boldsymbol{A}'_{\sigma\alpha} = \boldsymbol{A}_{\sigma}\boldsymbol{U}_{\alpha}, \quad \boldsymbol{B}'_{\delta\beta} = \boldsymbol{B}_{\delta}\boldsymbol{V}_{\beta}, \quad \boldsymbol{C}'_{\tau\gamma} = \boldsymbol{C}_{\tau}\boldsymbol{W}_{\gamma}.$$

Consequently,

$$\boldsymbol{Y} = \sum_{\sigma,\delta,\tau} \sum_{\alpha,\beta,\gamma} \boldsymbol{f}_{\sigma\delta\tau}\, \boldsymbol{g}_{\alpha\beta\gamma}\, \boldsymbol{A}'_{\sigma\alpha} \otimes\ \boldsymbol{B}'_{\delta\beta} \otimes\ \boldsymbol{C}'_{\tau\gamma}.$$

GENERAL RANK REDUCTION METHODS

Consider a general tensor $\boldsymbol{Z} = [z_{ijk}]$ with the mode sizes $\boldsymbol{n}$.

Tensor SVD (Higher Order SVD):

- Construct the unfolding matrices $\boldsymbol{Z}_1 = [z_{i,(jk)}], \quad \boldsymbol{Z}_2 = [z_{j,(ik)}], \quad \boldsymbol{Z}_3 = [z_{k,(ij)}]$.
- Perform rank revealing decompositions (via SVD)

$$\boldsymbol{Z}_1 = \boldsymbol{Q}_1\boldsymbol{R}_1 + \boldsymbol{E}_1, \quad \boldsymbol{Z}_2 = \boldsymbol{Q}_2\boldsymbol{R}_2 + \boldsymbol{E}_2, \quad \boldsymbol{Z}_3 = \boldsymbol{Q}_3\boldsymbol{R}_3 + \boldsymbol{E}_3,$$

  where

$$\boldsymbol{Q}_1 = [q^1_{i\alpha}], \quad \boldsymbol{Q}_2 = [q^2_{j\beta}], \quad \boldsymbol{Q}_3 = [q^3_{k\gamma}]$$

  are orthogonal $\boldsymbol{n} \times \boldsymbol{r}$ matrices.

- Compute the Tucker core tensor

$$h_{\alpha\beta\gamma} = \sum_{i,j,k} q^1_{i\alpha}\, q^2_{j\beta}\, q^3_{k\gamma}\, z_{ijk}.$$

- Finish with the Tucker approximation in the form

$$y_{ijk} \approx \widetilde{y}_{ijk} = \sum_{\alpha\beta\gamma} h_{\alpha\beta\gamma}\, q^1_{i\alpha}\, q^2_{j\beta}\, q^3_{k\gamma}.$$

COMPLEXITY $= O(n^4)$

GENERAL RANK REDUCTION METHODS

Alternating Least Squares (ALS):

- Freeze $\boldsymbol{Q_2}$, $\boldsymbol{Q_3}$ and compute

$$\boldsymbol{h^1_{i\beta\gamma} = \sum_{j,k} q^2_{j\beta}\, q^3_{k\gamma}\, z_{ijk}.}$$

  Consider a matrix $\boldsymbol{H_1 = [h^1_{i\beta\gamma}]}$ of size $\boldsymbol{n \times r^2}$ (here $\boldsymbol{\beta, \gamma}$ form a "long index" for columns) and find $\boldsymbol{Q_1}$ from a rank revealing decomposition

$$\boldsymbol{H_1 = Q_1 R_1 + F_1}$$

  with minimal $||\boldsymbol{F_1}||_{\boldsymbol{F}}$. Note that $\boldsymbol{Q_1}$ is a maximizer for $||\boldsymbol{Q}^\top \boldsymbol{H_1}||_{\boldsymbol{F}}$ over all matrices $\boldsymbol{Q}$ with $\boldsymbol{r}$ orthonormal columns.

- Freeze $\boldsymbol{Q_1}$, $\boldsymbol{Q_3}$ and find the best fit $\boldsymbol{Q_2}$.

- Freeze $\boldsymbol{Q_1}$, $\boldsymbol{Q_2}$ and find the best fit $\boldsymbol{Q_3}$.

- Repeat until convergence, then compute the Tucker core for the obtained mode frame matrices.

COMPLEXITY $= \boldsymbol{O(n^3 r + n^2 r^2 + n r^3)}$

GENERAL RECOMPRESSION ALGORITHM

Given $z_{ijk} = \sum_{\alpha,\beta,\gamma} h_{\alpha\beta\gamma}\ a_{i\alpha}\ b_{j\beta}\ c_{k\gamma}$ with the mode sizes n and mode ranks r_0, find its approximation of lesser rank with $r < r_0$.

- Compute orthogonal $n \times r_0$ matrices $Q_1 = [q^1_{i\alpha'}],\quad Q_2 = [q^2_{j\beta'}],\quad Q_3 = [q^3_{k\gamma'}]$ s.t.
$$[a_{i\alpha}] = Q_1 R_1,\quad [b_{j\beta}] = Q_2 R_2,\quad [c_{k\gamma}] = Q_3 R_3.$$
- Find an auxiliary $r_0 \times r_0 \times r_0$ tensor $h'_{\alpha'\beta'\gamma'} = \sum_{\alpha,\beta,\gamma} r^1_{\alpha'\alpha}\ r^2_{\beta'\beta}\ r^3_{\gamma'\gamma}\ h_{\alpha\beta\gamma}$ via 3 steps:
$$h^*_{\alpha'\beta\gamma} = \sum_{\alpha} r^1_{\alpha'\alpha}\ h_{\alpha\beta\gamma},\qquad h^{**}_{\alpha'\beta'\gamma} = \sum_{\beta} r^2_{\beta'\beta}\ h^*_{\alpha'\beta\gamma},\qquad h'_{\alpha'\beta'\gamma'} = \sum_{\gamma} r^2_{\gamma'\gamma}\ h^{**}_{\alpha'\beta'\gamma}.$$
- Reduce its mode ranks: $h'_{\alpha'\beta'\gamma'} \approx \sum_{\sigma=1}^{r}\sum_{\delta=1}^{r}\sum_{\tau=1}^{r} p^1_{\alpha'\sigma}\ p^2_{\beta'\delta}\ p^3_{\gamma'\tau}\widetilde{h}_{\sigma\delta\tau}.$
- Finally, $z_{ijk} \approx \sum_{\sigma=1}^{r}\sum_{\delta=1}^{r}\sum_{\tau=1}^{r} \widetilde{q}^1_{i\sigma}\ \widetilde{q}^2_{j\delta}\ \widetilde{q}^3_{l\tau}\ \widetilde{h}_{\sigma\delta\tau}$ with the Tucker factors
$$\widetilde{q}^1_{i\sigma} = \sum_{\alpha'=1}^{r_0} q^1_{i\alpha'} p^1_{\alpha'\sigma},\quad \widetilde{q}^2_{j\delta} = \sum_{\beta'=1}^{r_0} q^2_{j\beta'} p^2_{\beta'\delta},\quad \widetilde{q}^3_{j\tau} = \sum_{\gamma'=1}^{r_0} q^2_{k\gamma'} p^3_{\gamma'\tau}.$$

COMPLEXITY $= O(nr_0^2 + nr_0 r + r_0^4)$ (possibly $r_0^3 r$ instead of r_0^4)

TUCKER-TUCKER RECOMPRESSION ALGORITHM

- Find orthogonal $n \times r^2$ matrices $Q_1 = [q^1_{i,(\sigma'\alpha')}]$, $Q_2 = [q^2_{j,(\delta'\beta')}]$, $Q_3 = [q^3_{k,(\tau'\gamma')}]$ s.t.

$$[a_{i,(\sigma\alpha)}] = Q_1 R_1, \quad [b_{j,(\delta\beta)}] = Q_2 R_2, \quad [c_{k,(\tau\gamma)}] = Q_3 R_3.$$

- Define the auxiliary core tensor by

$$h_{\sigma'\alpha'\delta'\beta'\tau'\gamma'} = \sum_{\sigma,\delta,\tau} \sum_{\alpha,\beta,\gamma} r^1_{\sigma'\alpha'\sigma\alpha}\, r^2_{\delta'\beta'\delta\beta}\, r^3_{\tau'\gamma'\tau\gamma}\, f_{\sigma\delta\tau}\, g_{\alpha\beta\gamma}$$

and compute it through the following prescriptions:

$$\begin{aligned} h'_{\sigma'\alpha'\delta\beta\tau\gamma} &= \sum_{\sigma,\alpha} r^2_{\sigma'\alpha'\sigma\alpha}\, f_{\sigma\delta\tau}\, g_{\alpha\beta\gamma}, \\ h''_{\sigma'\alpha'\delta'\beta'\tau\gamma} &= \sum_{\delta,\beta} r^2_{\delta'\beta'\delta\beta}\, h'_{\sigma'\alpha'\delta\beta\tau\gamma}, \\ h_{\sigma'\alpha'\delta'\beta'\tau'\gamma'} &= \sum_{\tau,\gamma} r^3_{\tau'\gamma'\tau\gamma}\, h''_{\sigma'\alpha'\delta'\beta'\tau\gamma}. \end{aligned}$$

- Apply a mode rank reduction algorithm to the auxiliary tensor.
- Recompute the Tucker factors in the final Tucker approximation with mode ranks r.

COMPLEXITY $= O(nr^4 + r^8)$ **MEMORY** $= O(nr^2 + r^6)$

TUCKER-TUCKER ALS RECOMPRESSION ALGORITHM

- Freeze Q_2, Q_3 and find the best fit for Q_1: compute

$$v_{\beta'\delta\beta} = \sum_j q^2_{j\beta'}\, b_{j\delta\beta}, \quad w_{\gamma'\tau\gamma} = \sum_k q^3_{k\gamma'}\, c_{k\tau\gamma},$$

  then acquire

$$h_{i\beta'\gamma'} = \sum_{\sigma,\delta,\tau} \sum_{\alpha,\beta,\gamma} f_{\sigma\delta\tau}\, g_{\alpha\beta\gamma}\, v_{\beta'\delta\beta}\, w_{\gamma'\tau\gamma}\, a_{i\sigma\alpha}$$

  via the following prescriptions:

$$\begin{aligned} u'_{\alpha\beta\gamma'\tau} &= \sum_\gamma g_{\alpha\beta\gamma}\, w_{\gamma'\tau\gamma}, & u''_{\alpha\beta\gamma'\sigma\delta} &= \sum_\tau u'_{\alpha\beta\gamma'\tau}\, f_{\sigma\delta\tau}, \\ u_{\alpha\gamma'\sigma\beta'} &= \sum_{\delta,\beta} u''_{\alpha\beta\gamma'\sigma\delta}\, v_{\beta'\delta\beta}, & h_{i\beta'\gamma'} &= \sum_{\sigma,\alpha} u_{\alpha\gamma'\sigma\beta'}\, a_{i\sigma\alpha}. \end{aligned}$$

  Obtain Q_1 from a rank revealing decomposition of the matrix $H_1 = [h_{i,(\beta'\gamma')}]$ of size $n \times r^2$.

- Similarly, freeze Q_1, Q_3 and find the best fit for Q_2, then freeze Q_1, Q_2 and find the best fit for Q_3.

- Repeat until convergence.

COMPLEXITY $= O(nr^4 + r^6)$, MEMORY $= O(nr^2 + r^5)$.

BEWARE OF A POSSIBLE PITFALL

One may want to circumvent the computation of a full $r^2 \times r^2 \times r^2$ core and try to compress first the factor matrices via SVD (instead of QR).

This may not work!

EXAMPLE:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^\top = \begin{pmatrix} \boldsymbol{a} & \varepsilon^2 \boldsymbol{b} \end{pmatrix} \begin{pmatrix} \boldsymbol{c}^\top \\ \varepsilon^{-1} \boldsymbol{d}^\top \end{pmatrix} = \boldsymbol{a}\boldsymbol{c}^\top + \varepsilon \boldsymbol{b}\boldsymbol{d}^\top$$

$$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d} \text{ are unit-length vectors of size } \boldsymbol{n} \text{ and } (\boldsymbol{a}, \boldsymbol{b}) = (\boldsymbol{c}, \boldsymbol{d}) = 0$$

If we "compress" $\boldsymbol{U}$ and $\boldsymbol{V}$ separately, then the senior singular vectors of $\boldsymbol{U}$ and $\boldsymbol{V}$ would be $\boldsymbol{a}$ and $\boldsymbol{d}$. Hence,

$$\boldsymbol{A} \approx \gamma \boldsymbol{a}\boldsymbol{d}^\top,$$

which leaves us with no hope.

Thus, we have to compute the full core and treat all the factors *simultaneously*, not separately.

MATRIX INVERSION ALGORITHMS

Newton–Schultz method

$$X_{k+1} = 2X_k - X_k A X_k, \quad k = 0, 1, \ldots$$

converges quadratically (in exact arithmetics) if $||I - AX_0|| < 1$.

Let A be given in the Tucker format and the same structure be maintained for all iterates X_k:

$$Z_{k+1} = 2X_k - X_k A X_k, \quad X_{k+1} = \mathcal{P}(Z_{k+1}).$$

$\mathcal{P}$ is a (nonlinear) projector onto the manifold of matrices in the Tucker format with mode ranks r_1, r_2, r_3 selected and fixed before the method starts.

Truncated iteration converge still quadratically if

$$\mathcal{P}(A^{-1}) = A^{-1} + E, \quad ||E|| \leq \varepsilon$$

until $||X_k - A^{-1}|| > c\varepsilon$ for some $c > 1$.

GENERAL THEORY OF APPROXIMATE ITERATIONS

V a normed space

$B \in V$ the target of computaion

ITERATIVE PROCESS: $X_k = \Phi_k(X_{k-1})$

LEMMA.
Assume $\exists\ \alpha > 1,\ \varepsilon_\Phi,\ c_\Phi$ s.t.

$$||X - B|| \leq \varepsilon_\Phi \quad \Rightarrow$$

$$||\Phi_k(X) - B|| \ \leq\ c_\Phi ||X - B||^\alpha.$$

Then

$$||X_0 - B|| < \varepsilon \quad \Rightarrow$$

$$||X_k - B|| \ \leq\ c^{-1}\ (c\ ||X_0 - B||\ )^{\alpha^k}, \quad k = 1, 2\ldots$$

$$\varepsilon = \min\left(\varepsilon_\Phi,\ c^{-1}\right), \quad c = c_\Phi^{\frac{1}{\alpha-1}}$$

$\boldsymbol{S} \subset \boldsymbol{V}$ a subset of “structured” elements (e.g. structured matrices)

$\boldsymbol{R} : \boldsymbol{V} \longrightarrow \boldsymbol{S}$ a truncation operator.

$\boldsymbol{X} \in \boldsymbol{S} \quad \Rightarrow \quad \boldsymbol{R}(\boldsymbol{X}) = \boldsymbol{X}.$

TRUNCATED ITERATIVE PROCESS:

$$\boldsymbol{Y}_0 = \boldsymbol{R}(\boldsymbol{X}_0)$$

$$\boldsymbol{Y}_k = \boldsymbol{R}(\boldsymbol{\Phi}_k(\boldsymbol{Y}_{k-1}))$$

THEOREM 1.

Assume that

(1) Premises of Lemma are fulfilled.

(2) $R(B) = B$.

(3) $||X - B|| \le \varepsilon_\Phi \quad \Rightarrow \quad ||X - R(X)|| \le c_R \, ||X - B||$.

Then $\exists \ \delta > 0$ s.t.

$$Y_0 = R(Y_0), \quad ||Y_0 - B|| < \delta \quad \Rightarrow$$

$$||Y_k - B|| \ \le \ c_{R\Phi} \, ||Y_{k-1} - B||^\alpha, \quad k = 1, \, 2, \, \ldots$$

$$c_{R\Phi} = (c_R + 1) c_\Phi$$

W. Hackbusch, B.N. Khoromskij, E. Tyrtyshnikov. *Approximate iteration for structured matrices.* Preprint no. 112, Max-Planck-Institut für Mathematik in den Naturwissenschaften, Leipzig 2005. Numer. Math., DOI 10.1007/s00211-008-0143-0, 2008.

THEOREM 2.

Assume $\exists \quad \varepsilon_\Phi,\ c_R,\ \varepsilon_{RB}$ s.t

$$||X - B|| \le \varepsilon_\Phi \quad \Rightarrow$$
$$||X - R(X)|| \le c_R\, ||X - B|| + \varepsilon_{RB}.$$

Let m be the minimal k s.t.

$$e_{k-1}^\alpha \ \le\ \frac{\varepsilon_{RB}}{c_{R\Phi}}, \qquad c_{R\Phi} = (c_R + 1)c_\Phi.$$

Then errors $e_k = ||Y_k - B||$ of trun. iter. decrease superlinearly until $k \le m$:

$$k \le m - 1 \quad \Rightarrow \quad e_k \ \le\ 2c_{R\Phi}\, e_{k-1}^\alpha,$$

$$k \ge m \quad \Rightarrow \quad e_m \ \le\ 2\varepsilon_{RB}.$$

PROOF. $\quad Z_k := \Phi_k(Y_{k-1})$

$$||Y_k - B|| \le ||Y_k - Z_k|| + ||Z_k - B|| \le (c_R + 1)||Z_k - B|| + \varepsilon_{RB} \quad \Rightarrow$$

$$e_k \ \le\ c_{R\Phi}\, e_{k-1}^\alpha + \varepsilon_{RB} \ \le\ 2c_{R\Phi}\, e_{k-1}^\alpha.$$

MODIFIED NEWTON–SCHULTZ

$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k(2\boldsymbol{I} - \boldsymbol{A}\boldsymbol{X}_k) = \boldsymbol{X}_k(2\boldsymbol{I} - \boldsymbol{Y}_k)$ with $\boldsymbol{Y}_k = \boldsymbol{A}\boldsymbol{X}_k \quad \Rightarrow$ $\boldsymbol{A}\boldsymbol{X}_{k+1} = \boldsymbol{A}\boldsymbol{X}_k(2\boldsymbol{I} - \boldsymbol{Y}_k) \quad \Rightarrow$

$$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k(2\boldsymbol{I} - \boldsymbol{Y}_k), \quad \boldsymbol{X}_{k+1} = \boldsymbol{X}_k(2\boldsymbol{I} - \boldsymbol{Y}_k)$$

$$\boldsymbol{X}_0 \text{ is an initial guess for } \boldsymbol{A}^{-1}, \ \boldsymbol{Y}_0 = \boldsymbol{A}\boldsymbol{X}_0.$$

TRUNCATED VERSION:

$$\boldsymbol{H}_k = \mathcal{P}(2\boldsymbol{I} - \boldsymbol{Y}_k), \quad \boldsymbol{Y}_{k+1} = \mathcal{P}_1(\boldsymbol{Y}_k\boldsymbol{H}_k), \quad \boldsymbol{X}_{k+1} = \mathcal{P}_2(\boldsymbol{X}_k\boldsymbol{H}_k)$$

Projectors $\mathcal{P}_1, \mathcal{P}_2$ ought to maintain the required accuracy and $\mathcal{P}$ can be a way less accurate.

If $\boldsymbol{X}_0$ is close enough to $\boldsymbol{A}^{-1}$ then $\boldsymbol{Y}_k$ and $\boldsymbol{H}_k$ are close to $\boldsymbol{I}$, a “perfect structure” matrix. Experiments confirm that the modified Newton method is faster and more accurate than the standard Newton method.

INVERSION OF A 3D LAPLACIAN

$$\boldsymbol{A} = \triangle \otimes \boldsymbol{I} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \triangle \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{I} \otimes \triangle$$

$$\triangle = \mathbf{tridiag}(-1, 2, -1)$$

$N = n^3$	32^3	64^3	128^3	256^3
Time	**9** sec	**11** sec	**24** sec	**227** sec
$\|\|AX - I\|\|_F / \|\|I\|\|_F$	$3 \cdot 10^{-6}$	$4 \cdot 10^{-6}$	$2 \cdot 10^{-5}$	10^{-4}

For the Laplacian, projectors are chosen as follows:

$$\mathcal{P} = \mathcal{P}_{(2,2,2)}, \quad \mathcal{P}_1 = \mathcal{P}_2 = \mathcal{P}_{(12,12,12)}.$$

$\mathcal{P}_{(r_1 r_2 r_3)}$ is a projector onto tensors with mode ranks r_1, r_2, r_3.

INVERSION OF A NEWTON-POTENTIAL MATRIX

Mode ranks for the Newton potential matrix, $\varepsilon = 10^{-5}$

$N = n^3$	32^3	64^3	128^3	256^3
r	10	12	13	15

Timings for the Newton potential matrix

$N = n^3$	32^3	64^3	128^3	256^3
Time	**60** sec	**107** sec	**360** sec	**1574** sec
$\|\|AX - I\|\|_F / \|\|I\|\|_F$	10^{-2}	$9 \cdot 10^{-3}$	$5 \cdot 10^{-2}$	$4 \cdot 10^{-2}$

For the Newton potential matrix, to cause the method to converge we had to select

$$\mathcal{P} = \mathcal{P}_1 = \mathcal{P}_2 = \mathcal{P}_{(10,10,10)}.$$

SUBLINEAR COMPLEXITY FOR 2D TOEPLITZ INVERSION

Consider a 5-point Laplacian of order $N = n^2$ (2-level Toeplitz matrix). The inverse matrix is approximated by the Newton–Schultz method

$$X_{k+1} = \texttt{APPROXIMATION}(2X_k - X_k A X_k)$$

with a rank-structured approxination of all computed matrices: by matrices of limited tensor rank and limited displacement rank of each block.

n	64^2	128^2	256^2	512^2
Tensor rank A^{-1}	9	10	11	12
Averaged displacement rank of A^{-1}	13.5	13.5	16.8	18.6

Inversion of the 5-point Laplacian

Time behaves as $\mathcal{O}(\sqrt{N} r_{\text{mean}}^2)$,
r_{mean} = averaged displacement rank.

V.Olshevsky, I.Oseledets, E.Tyrtyshnikov,
Superfast inversion of two-level Toeplitz matrices using Newton iteration and tensor-displacement structure, *Operator Theory Advances and Applications*, vol. 179, pp. 229–240 (2007).

Low-parametric representations of inverse matrices contain $o(N)$ parameters.

Hence, all difficulties are relegated to the representation of vectors, not matrices!

TENSOR STRUCTURE IN VECTORS

$$\boldsymbol{x} = \begin{bmatrix} \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \\ \mathbf{7} \\ \mathbf{8} \\ \mathbf{9} \end{bmatrix} \qquad \Leftrightarrow \qquad \boldsymbol{X} = \begin{bmatrix} \mathbf{1} & \mathbf{4} & \mathbf{7} \\ \mathbf{2} & \mathbf{5} & \mathbf{8} \\ \mathbf{3} & \mathbf{6} & \mathbf{9} \end{bmatrix}$$

$$\boldsymbol{X} = \mathtt{MATRIX}(\boldsymbol{x}) \qquad \Leftrightarrow \qquad \boldsymbol{x} = \mathtt{VECTOR}(\boldsymbol{X})$$

$$\boldsymbol{X} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 2 \end{bmatrix} = \boldsymbol{u}_1 \boldsymbol{v}_1^\top + \boldsymbol{u}_2 \boldsymbol{v}_2^\top$$

$$\boldsymbol{x} = \boldsymbol{v}_1 \otimes \boldsymbol{u}_1 + \boldsymbol{v}_2 \otimes \boldsymbol{u}_2$$

EIGENVECTOR STRUCTURE

$$\boldsymbol{M} = \boldsymbol{A} \otimes \boldsymbol{B} + \boldsymbol{C} \otimes \boldsymbol{D}$$

If $\boldsymbol{A}$, $\boldsymbol{C}$ and $\boldsymbol{B}$, $\boldsymbol{D}$ are two pairs of commuting matrices, then any eigenvector has a tensor rank-1 structure.

DISCRETE LAPLACIAN CASE

$$\boldsymbol{M}\boldsymbol{u} = \boldsymbol{\lambda}\boldsymbol{u}, \quad \boldsymbol{M} = \boldsymbol{A} \otimes \boldsymbol{I} + \boldsymbol{I} \otimes \boldsymbol{A}$$

$$\boldsymbol{A} = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \cdots & \cdots & \cdots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}$$

$$\boldsymbol{x}_{kl} = \boldsymbol{u}^k \otimes \boldsymbol{v}^l \qquad u_s^k = \sin \frac{\pi k s}{n+1}, \qquad v_t^l = \sin \frac{\pi l t}{n+1}$$

$$\lambda_{kl} = 4\sin^2 \frac{\pi k}{2(n+1)} + 4\sin^2 \frac{\pi l}{2(n+1)}, \quad 1 \le k, l \le n$$

USE TENSOR VECTORS IN EIGENSOLVERS

LANCZOS:

- Choose an initial vector $\boldsymbol{p_1}$ with $||\boldsymbol{p_1}|| = \mathbf{1}$ and set $\boldsymbol{p_0} = \mathbf{0}$, $\boldsymbol{b_0} = \mathbf{0}$.
- For $\boldsymbol{k = 1, 2, \ldots}$ compute

$$\begin{aligned}
z_k &= M p_k \\
a_k &= (z_k, p_k) \\
q_k &= z_k - a_k p_k - b_{k-1} p_{k-1} \\
b_k &= ||q_k|| \\
p_{k+1} &= q_k / b_k
\end{aligned}$$

- Compute the Rietz values as the eigenvalues of a projection $\boldsymbol{k \times k}$ matrix (a symmetric tridiagonal matrix consisting of the values $\boldsymbol{a_k, b_k}$)

$$M_k = P_k^\top M P_k, \quad P_k = [p_1, \ldots, p_k].$$

TENSOR LANCZOS

- Choose an initial vector $\boldsymbol{p_1}$ with $||\boldsymbol{p_1}|| = \mathbf{1}$ and set $\boldsymbol{p_0} = \mathbf{0}$, $\boldsymbol{b_0} = \mathbf{0}$.
- For $\boldsymbol{k} = \mathbf{1}, \mathbf{2}, \ldots$ compute

$$
\begin{aligned}
\boldsymbol{z_k} &= \boldsymbol{M p_k} \\
\boldsymbol{a_k} &= (\boldsymbol{z_k}, \boldsymbol{p_k}) \\
\boldsymbol{q_k} &= \boldsymbol{T_\varepsilon}(\boldsymbol{z_k} - \boldsymbol{a_k p_k} - \boldsymbol{b_{k-1} p_{k-1}}) \\
\boldsymbol{b_k} &= ||\boldsymbol{q_k}|| \\
\boldsymbol{p_{k+1}} &= \boldsymbol{q_k}/\boldsymbol{b_k}
\end{aligned}
$$

- Compute the Rietz values using $\boldsymbol{k} \times \boldsymbol{k}$ projection matrices.

STANDARD VERSUS TENSOR (50 iterations)

n	1000	2000	4000	6000
Lanczos time (sec)	2.8	12.1	76.7	224.9
Tensor Lanczos time (sec)	0.4	0.7	1.5	2.2

For $\boldsymbol{n} = \mathbf{6000}$ we observe a $\mathbf{100}$ times acceleration.

MORE EXAMPLES

$$\boldsymbol{M_r = M - \sum_{t=1}^{\rho} D_t \otimes D_t}$$

$\boldsymbol{M} = -$Laplacian. $\boldsymbol{D_t}$ are diagonal matrices with positive entries.

Compare maximal eigenvalues on the 50th iteration.
$\boldsymbol{n = 300^2}$, truncation rank $= \mathbf{10}$, $\boldsymbol{\varepsilon} = \mathbf{10^{-2}}$.

$\boldsymbol{\rho}$	1	3	5	7	9
Standard Lanczos	7.989	7.957	7.925	7.900	7.893
Tensor Lanczos	7.977	7.940	7.917	7.893	7.906

Entries of $\boldsymbol{D_t}$ are uniform grid values of $\boldsymbol{(1 + T_t(x))/10}$,
$\boldsymbol{T_t}$ is the Chebyshev polynomial of degree $\boldsymbol{t}$.

$\boldsymbol{\rho}$	1	3	5	7	9
Standard Lanczos	7.862	7.615	7.302	6.800	6.460
Tensor Lanczos	7.852	7.608	7.292	6.789	6.452

Entries of $\boldsymbol{D_t}$ come from random vectors with uniform distribution on $[\mathbf{0}, \mathbf{1}]$.

SUBLINEAR COMPLEXITY

For 2-dimensional problems time grows as SQUARE ROOT of total number of nodes!

For d-dimensional problems time should grow as ROOT OF DEGREE d of total number of nodes!

FUTURE RESEARCH

- Additional structure for matrix factors (wavelet sparsification, Toeplitz-like structure, circulant-plus-low rank structure).

- Iterative methods for matrix functions (sign functions, square root, matrix exponential).

- Iterative methods for linear systems with structured vectors (PCG,BiCGStab) and preconditioners computed by the Newton method.

- Eigenvalue solvers with tensor-structured eigenvectors.

FUTURE RESEARCH

A challenge topic for the linear algebra and matrix analysis community:

Which properties of matrices do account for a Tucker approximation with low mode ranks for the inverses, matrix exponentials and other matrix functions?

All experiments are in favour of the following hypothesis: for all practical operators of mathematical physics (differential operators, integral operators with smooth, singular and hypersingular kernels) on tensor grids, standard matrix functions can be approximated in the Tucker format with ranks of order

$$\boldsymbol{r \sim \log^{\alpha} n \log^{\beta} \varepsilon^{-1},}$$

with some constants $\boldsymbol{\alpha, \beta}$. The known theorems on tensor structure of the matrix functions rely on analytical considerations. Matrix-grounds attempts are very few:

I.V. Oseledets, E.E. Tyrtyshnikov, and N.L. Zamarashkin, Matrix inversion cases with size-independent tensor rank estimates, *Linear Algebra Appl.*, submitted, 2008.

E.E. Tyrtyshnikov, Tensor Ranks for Inversion of Tensor-Product Binomials, submitted, 2008.