Orthogonalization schemes in tensor train format

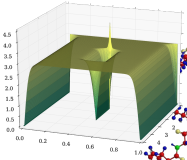
29th Biennial Numerical Analysis Conference,
June 28th 2023, Glasgow

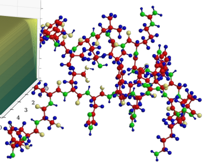
Martina Iannacito*
joint work with Olivier Coulaud† and Luc Giraud†

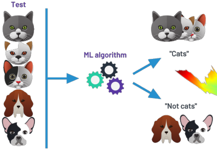
* KU Leuven
† Centre Inria at the University of Bordeaux

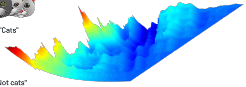
The context in matrix computation

- Stochastic equations
- Uncertainty quantification problems
- Quantum and vibration chemistry
- Optimization
- Machine learning









reduce their problems to

Least-squares	Eigenpairs	Linear systems
$\min_{x \in \mathcal{S}} \|\|b - Ax\|\|$	$Ax = \lambda x$	$Ax = b$

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1,$$

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1 \rangle}{||q_1||^2} q_1,$$

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1 \rangle}{||q_1||^2} q_1, \ldots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j \rangle}{||q_j||^2} q_j$$

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1 \rangle}{||q_1||^2} q_1, \dots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j \rangle}{||q_j||^2} q_j$$

How to make an algorithm out of it?

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1 \rangle}{||q_1||^2} q_1, \ldots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j \rangle}{||q_j||^2} q_j$$

How to make an algorithm out of it?

let $Q_k = [q_1, \ldots, q_k]$ be a $n \times k$ matrix

All projections, then subtractions

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1\rangle}{||q_1||^2} q_1, \ldots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j\rangle}{||q_j||^2} q_j$$

How to make an algorithm out of it?

let $Q_k = [q_1, \ldots, q_k]$ be a $n \times k$ matrix

All projections, then subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$$

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1 \rangle}{||q_1||^2} q_1, \ldots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j \rangle}{||q_j||^2} q_j$$

How to make an algorithm out of it?

let $Q_k = [q_1, \ldots, q_k]$ be a $n \times k$ matrix

All projections, then subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$$

Projections alternate subtractions

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1 \rangle}{||q_1||^2} q_1, \dots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j \rangle}{||q_j||^2} q_j$$

How to make an algorithm out of it?

let $Q_k = [q_1, \dots, q_k]$ be a $n \times k$ matrix

All projections, then subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$$

Projections alternate subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$$

From theory to coding: an example I

Gram-Schmidt process [Leon et al. 2013]

Given $\{a_h\}_h$ linearly independent vectors, we construct an orthogonal basis $\{q_h\}$ such that

$$q_1 = a_1, \quad q_2 = a_2 - \frac{\langle a_2, q_1\rangle}{||q_1||^2} q_1, \dots \quad q_{k+1} = a_{k+1} - \sum_{j=1}^{k} \frac{\langle a_k, q_j\rangle}{||q_j||^2} q_j$$

How to make an algorithm out of it?

let $Q_k = [q_1, \dots, q_k]$ be a $n \times k$ matrix

All projections, then subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$$

Projections alternate subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$$

Are these versions equivalent in finite precision arithmetic?

From theory to coding: an example II

Not equivalent in general

Classical Gram-Schmidt	Modified Gram-Schmidt
All projections, then subtractions	Projections alternate subtractions
$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$	$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \ldots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$

From theory to coding: an example II

Not equivalent in general

Classical Gram-Schmidt	Modified Gram-Schmidt
All projections, then subtractions	Projections alternate subtractions
$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$	$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} \qquad \text{with} \qquad \varepsilon = 1e-10 \qquad \text{in fp64}$$

From theory to coding: an example II

Not equivalent in general

Classical Gram-Schmidt	Modified Gram-Schmidt
All projections, then subtractions	Projections alternate subtractions
$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$	$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} \qquad \text{with} \qquad \varepsilon = 1e-10 \qquad \text{in fp64}$$

$$Q_{\text{CGS}} = \begin{bmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix},$$

From theory to coding: an example II

Not equivalent in general

Classical Gram-Schmidt	Modified Gram-Schmidt
All projections, then subtractions	Projections alternate subtractions
$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$	$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} \qquad \text{with} \qquad \varepsilon = 1e-10 \qquad \text{in fp64}$$

$$Q_{\text{CGS}} = \begin{bmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix}, \ \langle q_2, q_3 \rangle = \frac{1}{2}$$

From theory to coding: an example II

Not equivalent in general

Classical Gram-Schmidt

All projections, then subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$$

Modified Gram-Schmidt

Projections alternate subtractions

$$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} \quad \text{with} \quad \varepsilon = 1e-10 \quad \text{in fp64}$$

$$Q_{\text{CGS}} = \begin{bmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix}, \ \langle q_2, q_3 \rangle = \frac{1}{2}$$

$$Q_{\text{MGS}} = \begin{bmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon/2 \\ 0 & \varepsilon & -\varepsilon/2 \\ 0 & 0 & \varepsilon \end{bmatrix},$$

Not equivalent in general

Classical Gram-Schmidt	Modified Gram-Schmidt
All projections, then subtractions	Projections alternate subtractions
$q_{k+1} \leftarrow (\mathbb{I}_n - Q_k Q_k^\top) a_{k+1}$	$q_{k+1} \leftarrow (\mathbb{I}_n - q_k \otimes q_k) \dots (\mathbb{I}_n - q_1 \otimes q_1) a_{k+1}$

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} \qquad \text{with} \qquad \varepsilon = 1e-10 \qquad \text{in fp64}$$

$$Q_{\text{CGS}} = \begin{bmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{bmatrix}, \ \langle q_2, q_3 \rangle = \frac{1}{2}$$

$$Q_{\text{MGS}} = \begin{bmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon/2 \\ 0 & \varepsilon & -\varepsilon/2 \\ 0 & 0 & \varepsilon \end{bmatrix}, \ \langle q_1, q_3 \rangle = \varepsilon \sqrt{\frac{2}{3}}$$

This example was taken from [Björck 1996].

Loss of orthogonality

Let $Q_k = [q_1, \ldots, q_k]$ be the computed orthogonal basis, then its **loss of orthogonality** is

$$||\mathbb{I}_k - Q_k^\top Q_k||.$$

It measures the quality in terms of orthogonality of the computed basis. It is linked with the computational precision u and the linearly dependency of the input vectors $A_k = [a_1, \ldots, a_k]$, estimated through $\kappa(A_k)$.

Matrix		
Source	*Algorithm*	$\left\|\mathbb{I}_k - Q_k^\top Q_k\right\|$
[Stathopoulos et al. 2002]	**Gram**	$\mathcal{O}(u\kappa^2(A_k))$
[Giraud et al. 2005]	**CGS**	$\mathcal{O}(u\kappa^2(A_k))$
[Björck 1967]	**MGS**	$\mathcal{O}(u\kappa(A_k))$
[Giraud et al. 2005]	**CGS2**	$\mathcal{O}(u)$
[Giraud et al. 2005]	**MGS2**	$\mathcal{O}(u)$
[Wilkinson 1965]	**Householder**	$\mathcal{O}(u)$

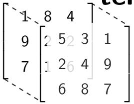
The advent of tensors

matrix

$$\begin{bmatrix} 1 & 8 & 4 \\ 9 & 2 & 2 \\ 7 & 1 & 6 \end{bmatrix}$$

since the last century

tensor



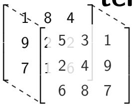
The advent of tensors

matrix

$$\begin{bmatrix} 1 & 8 & 4 \\ 9 & 2 & 2 \\ 7 & 1 & 6 \end{bmatrix}$$

since the last century

tensor



+ Better representation
- Curse of dimensionality

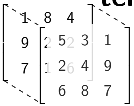
The advent of tensors

matrix

$$\begin{bmatrix} 1 & 8 & 4 \\ 9 & 2 & 2 \\ 7 & 1 & 6 \end{bmatrix}$$

since the last century

tensor



+ Better representation
– Curse of dimensionality

Approximation techniques were proposed

- Canonical Polyadic
- Tucker
- Hierarchical Tucker
- Tensor-Train

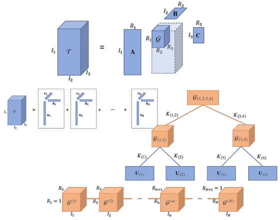


Figure: from [Bi et al. 2022]

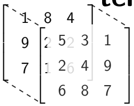
The advent of tensors

matrix

$$\begin{bmatrix} 1 & 8 & 4 \\ 9 & 2 & 2 \\ 7 & 1 & 6 \end{bmatrix}$$

since the last century

tensor



+ Better representation
- Curse of dimensionality

Approximation techniques were proposed

- Canonical Polyadic
- Tucker
- Hierarchical Tucker
- Tensor-Train

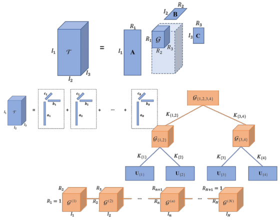
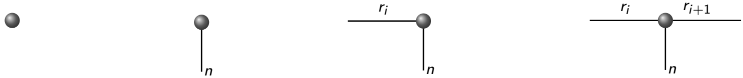


Figure: from [Bi et al. 2022]

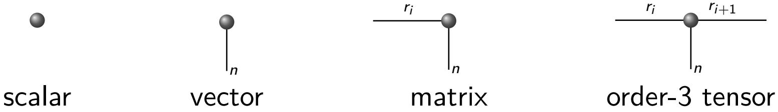
Approximation techniques introduce **norm-wise** compression errors

Tensor Train formalism



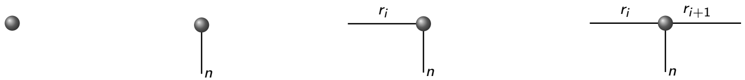
scalar vector matrix order-3 tensor

Tensor Train formalism



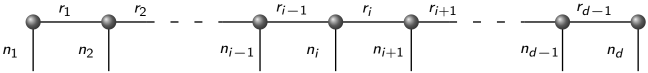
Let $\mathbf{x} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be order-d tensor, its TT-representation is

Tensor Train formalism



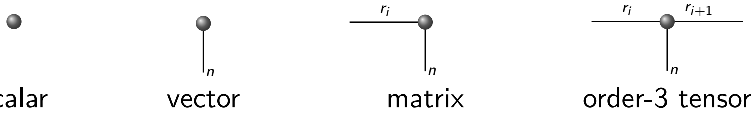
scalar | vector | matrix | order-3 tensor

Let $\mathbf{x} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be order-d tensor, its TT-representation is

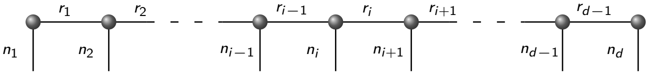


+ The storage cost is $\mathcal{O}(dnr^2)$ with $r = \max\{r_i\}$, said **TT-ranks**

Tensor Train formalism

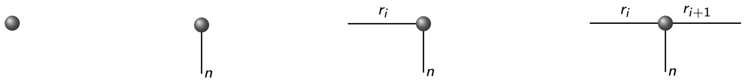


Let $\mathbf{x} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be order-d tensor, its TT-representation is



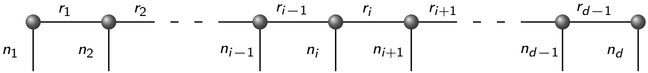
+ The storage cost is $\mathcal{O}(dnr^2)$ with $r = \max\{r_i\}$, said **TT-ranks**
- Linear combinations increase the TT-ranks

Tensor Train formalism



scalar vector matrix order-3 tensor

Let $\mathbf{x} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be order-d tensor, its TT-representation is



+ The storage cost is $\mathcal{O}(dnr^2)$ with $r = \max\{r_i\}$, said **TT-ranks**
- Linear combinations increase the TT-ranks

TT-rounding [Oseledets 2011]

If $\mathbf{z}$ is an order-d tensor in TT-format and $\delta \in (0, 1)$, then $\bar{\mathbf{z}} = \texttt{TT-rounding}(\mathbf{z}, \delta)$ such that

$$\|\mathbf{z} - \bar{\mathbf{z}}\| \leq \delta \|\mathbf{z}\|$$

The questions

In the Tensor-Train framework

compression precision δ

- norm-wise perturbation
- TT-model [Oseledets 2011]

computational precision u

- component-wise perturbation
- IEEE model [Higham 2002]

The questions

In the Tensor-Train framework

compression precision δ	computational precision u
• norm-wise perturbation	• component-wise perturbation
• TT-model [Oseledets 2011]	• IEEE model [Higham 2002]

- How to design orthogonalization kernels for tensor subspace?
- Does compression affect the loss of orthogonality?
- Are tensor results related with the known linear algebra ones?

Classical and Modified Gram-Schmidt

$\mathcal{Q}, R = \texttt{CGS}(\mathcal{A}, \delta)$

Input: $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$, $\delta \in \mathbb{R}_+$

for $i = 1, \ldots, m$ **do**
- $\mathbf{p} = \mathbf{a}_i$
- **for** $j = 1, \ldots, i-1$ **do**
  - $R(i,j) = \langle \mathbf{a}_i, \mathbf{q}_j \rangle$
  - $\mathbf{p} = \mathbf{p} - R(i,j)\mathbf{q}_j$
- **end**
- $\mathbf{p} = \texttt{TT-rounding}(\mathbf{p}, \delta)$
- $R(i,i) = ||\mathbf{p}||$
- $\mathbf{q}_i = 1/R(i,i)\,\mathbf{p}$

end

Output: $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, R

$\mathcal{Q}, R = \texttt{MGS}(\mathcal{A}, \delta)$

Input: $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$, $\delta \in \mathbb{R}_+$

for $i = 1, \ldots, m$ **do**
- $\mathbf{p} = \mathbf{a}_i$
- **for** $j = 1, \ldots, i-1$ **do**
  - $R(i,j) = \langle \mathbf{p}, \mathbf{q}_j \rangle$
  - $\mathbf{p} = \mathbf{p} - R(i,j)\mathbf{q}_j$
- **end**
- $\mathbf{p} = \texttt{TT-rounding}(\mathbf{p}, \delta)$
- $R(i,i) = ||\mathbf{p}||$
- $\mathbf{q}_i = 1/R(i,i)\,\mathbf{p}$

end

Output: $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, R

They readily write in TT-format.

CGS and MGS with reorthogonalization

$\mathcal{Q}, R = \texttt{CGS2}(\mathcal{A}, \delta)$

Input: $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$, $\delta \in \mathbb{R}_+$

for $i = 1, \ldots, m$ **do**
  $\mathbf{p}_k = \mathbf{a}_i$
  for $k = 1, 2$ **do**
    $\mathbf{p}_k = \mathbf{p}_{k-1}$
    for $j = 1, \ldots, i-1$ **do**
      $R(i,j) = \langle \mathbf{p}_{k-1}, \mathbf{q}_j \rangle$
      $\mathbf{p}_k = \mathbf{p}_k - R(i,j)\mathbf{q}_j$
    end
    $\mathbf{p}_k = \texttt{TT-rounding}(\mathbf{p}_k, \delta)$
  end
  $R(i,i) = ||\mathbf{p}_2||$
  $\mathbf{q}_i = 1/R(i,i)\,\mathbf{p}_2$
end

Output: $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, R

$\mathcal{Q}, R = \texttt{MGS2}(\mathcal{A}, \delta)$

Input: $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$, $\delta \in \mathbb{R}_+$

for $i = 1, \ldots, m$ **do**
  $\mathbf{p}_k = \mathbf{a}_i$
  for $k = 1, 2$ **do**
    $\mathbf{p}_k = \mathbf{p}_{k-1}$
    for $j = 1, \ldots, i-1$ **do**
      $R(i,j) = \langle \mathbf{p}_{k}, \mathbf{q}_j \rangle$
      $\mathbf{p}_k = \mathbf{p}_k - R(i,j)\mathbf{q}_j$
    end
    $\mathbf{p}_k = \texttt{TT-rounding}(\mathbf{p}_k, \delta)$
  end
  $R(i,i) = ||\mathbf{p}_2||$
  $\mathbf{q}_i = 1/R(i,i)\,\mathbf{p}_2$
end

Output: $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, R

Gram approach - matrix format

Let $A = [a_1, \ldots, a_m]$, then we look for $A = QR$ with $Q^\top Q = \mathbb{I}_m$ compute the Gram matrix

$$A^\top A = (R^\top Q^\top) QR = R^\top R$$

this is (almost) the **Cholesky** factorization of $A^\top A$ that can be written as

$$A^\top A = R^\top R = LL^\top$$

with the Cholesky factor $L = R^\top$. Thus, it follows

$$A = QR = QL^\top$$

from which we obtain Q by solving a linear system.

Gram approach - TT-format

$\mathcal{Q}, R = \texttt{Gram}(\mathcal{A}, \delta)$

Input: $\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_m\}$, $\delta \in \mathbb{R}_+$

G is $(m \times m)$ Gram matrix from $\mathcal{A}$

$L = \texttt{cholesky}(G)$

$\{\mathbf{p}_1, \ldots, \mathbf{p}_m\}$ from $\mathcal{A}$ and $(L^\top)^{-1}$

for $i = 1, \ldots, m$ **do**

  $\mathbf{q}_i = \texttt{TT-rounding}(\mathbf{p}_i, \delta)$

end

Output: $\mathcal{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, R

In TT-format the following modifications occur

- $G(i, j)$ is the scalar product of $\mathbf{a}_i$ and $\mathbf{a}_j$
- The inverse of $L^\top$ is explicitly computed
- $\mathbf{p}_i$ is constructed as a linear combination of $\mathcal{A}$ elements

Householder transformation

Given a vector $x \in \mathbb{R}^n$ and a direction $y \in \mathbb{R}^n$, the Householder reflector H reflects x along y, i.e.,

$$Hx = ||x||y \qquad \text{with} \qquad ||y|| = 1.$$

Thanks to its properties, H writes as

$$H = \mathbb{I}_n - \frac{2}{||z||^2} z \otimes z \qquad \text{with} \qquad z = (x - ||x||y).$$

The practical implementation of the Householder transformation kernel uses the components of the input vectors.

Remark

The Householder algorithm does **not** readily apply to tensor in TT-formats, because of the compressed nature of this format. It is described in [Iannacito 2022]

Numerical experiments in TT-format

Let $\mathbf{\Delta}_d$ is be the TT-Laplacian, then $\{\mathbf{a}_k\}$ are ‘Krylov tensors’, i.e.,

$$\mathbf{a}_{k+1} = \texttt{TT-rounding}(\mathbf{\Delta}_d \mathbf{x}_k, \texttt{max\_rank} = 1) \quad \text{with} \quad \mathbf{x}_{k+1} = \frac{1}{\|\mathbf{a}_{k+1}\|}\mathbf{a}_{k+1}$$

Numerical experiments in TT-format

Let $\mathbf{\Delta}_d$ is be the TT-Laplacian, then $\{\mathbf{a}_k\}$ are 'Krylov tensors', i.e.,

$$\mathbf{a}_{k+1} = \texttt{TT-rounding}(\mathbf{\Delta}_d \mathbf{x}_k, \texttt{max\_rank} = 1) \quad \text{with} \quad \mathbf{x}_{k+1} = \frac{1}{\|\mathbf{a}_{k+1}\|}\mathbf{a}_{k+1}$$

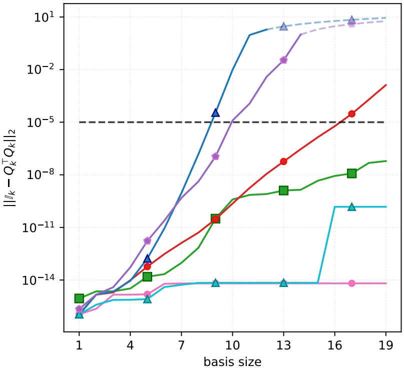


Figure: 20 tensors of order $d = 3$ and mode size $n = 15$, compression precision $\delta = 10^{-5}$, computational precision $u = \mathcal{O}(10^{-16})$

- **Gram approach**
- **CGS**
- **MGS**
- **CGS2**
- **MGS2**
- **Householder transformation**

Loss of orthogonality for $\{\mathbf{a}_k\}$ 'Krylov tensors'

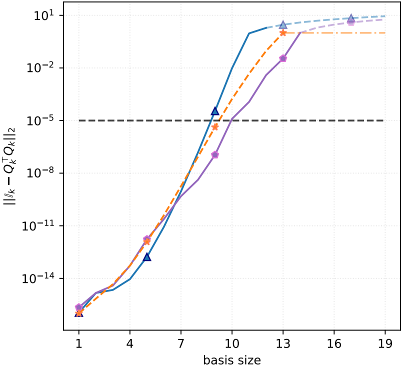


Figure: 20 tensors of order $d = 3$ and mode size $n = 15$, compression precision $\delta = 10^{-5}$, computational precision $u = \mathcal{O}(10^{-16})$

- **Gram approach**
- **CGS**
- $\delta\kappa^2(A_k)$ with

$$A_k = [\mathtt{vec}(\mathbf{a}_1), \ldots \mathtt{vec}(\mathbf{a}_k)]$$

then we conjecture

$$\mathcal{O}(\delta\kappa^2(A_k))$$

Loss of orthogonality for $\{\mathbf{a}_k\}$ 'Krylov tensors'

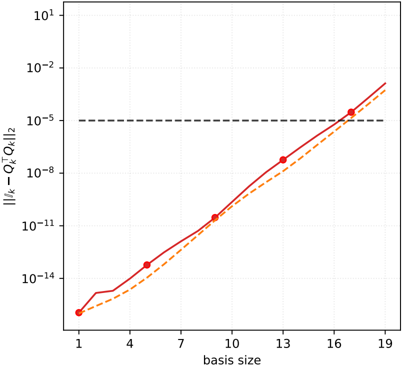


Figure: 20 tensors of order $d = 3$ and mode size $n = 15$, compression precision $\delta = 10^{-5}$, computational precision $u = \mathcal{O}(10^{-16})$

- **MGS**
- $\delta\kappa(A_k)$ with

$$A_k = [\mathtt{vec}(\mathbf{a}_1), \ldots \mathtt{vec}(\mathbf{a}_k)]$$

then we conjecture

$$\mathcal{O}(\delta\kappa(A_k))$$

Loss of orthogonality for $\{\mathbf{a}_k\}$ 'Krylov tensors'

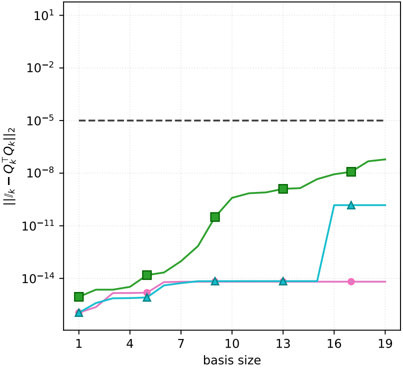


Figure: 20 tensors of order $d = 3$ and mode size $n = 15$, compression precision $\delta = 10^{-5}$, computational precision $u = \mathcal{O}(10^{-16})$

- **CGS2**
- **MGS2**
- **Householder transformation**

then we conjecture

$$\mathcal{O}(\delta)$$

Loss of orthogonality: matrix vs tensor

Algorithm	Matrix, theoretical $\left\lVert \mathbb{I}_k - Q_k^\top Q_k \right\rVert$	TT-format, conjecture $\left\lVert \mathbb{I}_k - \mathcal{Q}_k^\top \mathcal{Q}_k \right\rVert$
Gram	$\mathcal{O}(u\kappa^2(A_k))$	$\mathcal{O}(\delta\kappa^2(\mathcal{A}_k))$
CGS	$\mathcal{O}(u\kappa^2(A_k))$	$\mathcal{O}(\delta\kappa^2(\mathcal{A}_k))$
MGS	$\mathcal{O}(u\kappa(A_k))$	$\mathcal{O}(\delta\kappa(\mathcal{A}_k))$
CGS2	$\mathcal{O}(u)$	$\mathcal{O}(\delta)$
MGS2	$\mathcal{O}(u)$	$\mathcal{O}(\delta)$
Householder	$\mathcal{O}(u)$	$\mathcal{O}(\delta)$

with u the computational precision, δ the compression precision

Conclusions and perspectives

- All the 6 kernels can be generalized to the TT-framework
- Loss of orthogonality bounds appears to hold true with
  - > the compression precision δ independent from the machine architecture
  - > the compression precision δ replacing the computational one u
  - > the compression acting norm-wise rather than component-wise

More detailed results can be found at [Coulaud et al. 2022]

Conclusions and perspectives

- All the 6 kernels can be generalized to the TT-framework
- Loss of orthogonality bounds appears to hold true with
  - the compression precision δ independent from the machine architecture
  - the compression precision δ replacing the computational one u
  - the compression acting norm-wise rather than component-wise

More detailed results can be found at [Coulaud et al. 2022]

What is left out?

- Theoretical proof of the loss of orthogonality bounds
- Investigating the quality of the tensor subspace spanned by the orthogonal basis

Computational costs: matrix vs tensor

	cost in fp operations	cost in TT-rounding
Gram	$\mathcal{O}(2nm^2)$	m
CGS	$\mathcal{O}(2nm^2)$	m
MGS	$\mathcal{O}(2nm^2)$	m
CGS2	$\mathcal{O}(4nm^2)$	$2m$
MGS2	$\mathcal{O}(4nm^2)$	$2m$
Householder	$\mathcal{O}(2nm^2 - 2m^3/3)$	$4m$

Bibliography

[1] Y. Bi et al. "Chapter 1 - Tensor decompositions: computations, applications, and challenges". In: *Tensors for Data Processing*. Ed. by Y. Liu. Academic Press, 2022, pp. 1–30. ISBN: 978-0-12-824447-0. DOI: 10.1016/B978-0-12-824447-0.00007-8.

[2] Å. Björck. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, 1996. DOI: 10.1137/1.9781611971484.

[3] Å. Björck. "Solving linear least squares problems by Gram-Schmidt orthogonalization". In: *BIT Numerical Mathematics* 7.1 (Mar. 1967), pp. 1–21. DOI: 10.1007/BF01934122.

[4] O. Coulaud, L. Giraud, and M. Iannacito. *On some orthogonalization schemes in Tensor Train format*. Tech. rep. RR-9491. Inria Bordeaux - Sud-Ouest, Nov. 2022.

[5] L. Giraud, J. Langou, and M. Rozložník. "The loss of orthogonality in the Gram-Schmidt orthogonalization process". In: *Computers & Mathematics with Applications* 50.7 (2005). Numerical Methods and Computational Mechanics, pp. 1069–1075. DOI: 10.1016/j.camwa.2005.08.009.

[6] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. Second. Society for Industrial and Applied Mathematics, 2002. DOI: 10.1137/1.9780898718027.

[7] M. Iannacito. "Numerical linear algebra and data analysis in large dimensions using tensor format". Theses. Université de Bordeaux, Dec. 2022.

[8] S. J. Leon, Å. Björck, and W. Gander. "Gram-Schmidt orthogonalization: 100 years and more". In: *Numerical Linear Algebra with Applications* 20.3 (2013), pp. 492–532. DOI: 10.1002/nla.1839.

[9] I. V. Oseledets. "Tensor-Train Decomposition". In: *SIAM Journal on Scientific Computing* 33.5 (2011), pp. 2295–2317. DOI: 10.1137/090752286.

[10] A. Stathopoulos and K. Wu. "A Block Orthogonalization Procedure with Constant Synchronization Requirements". In: *SIAM Journal on Scientific Computing* 23.6 (2002), pp. 2165–2182. DOI: 10.1137/S1064827500370883.

[11] J. H. Wilkinson. *The algebraic eigenvalue problem*. en. Numerical Mathematics and Scientific Computation. Oxford, England: Clarendon Press, 1965.

Thanks for the attention.

Questions?