Tensor Train Approximations

Riemannian Methods, Randomized Linear Algebra and Applications to Machine Learning

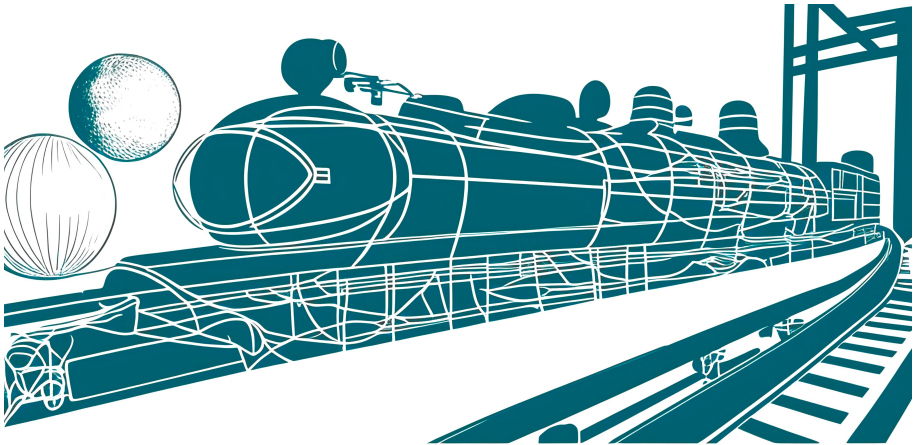
Thesis defense of **Willem Hendrik (Rik) Voorhaar** for obtaining the title of Doctorat ès sciences
Thesis advisor: Prof. **Bart Vandereycken**

Jury: Dr. Gilles Vilmart, Prof. Lars Grasedyck, Prof. Sergey Dolgov

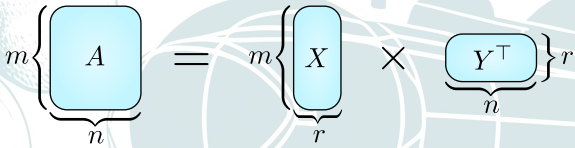
UNIVERSITÉ DE GENÈVE

FACULTÉ DES SCIENCES
Section de mathématiques

Part 1 | Low-rank matrices



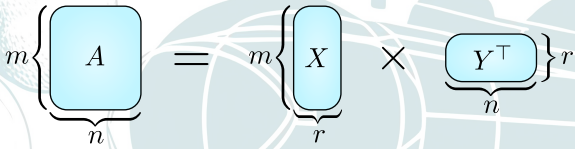
A matrix is low-rank if it can be written as a product of two smaller matrices.



A has rank $\leq r \quad \Leftrightarrow \quad A = XY^\top$ with X size $m \times r$ and Y size $n \times r$.

§1 | Low-rank matrices

A matrix is low-rank if it can be written as a product of two smaller matrices.

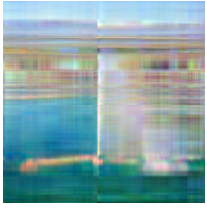


A has rank $\leq r \quad \Leftrightarrow \quad A = XY^\top$ with X size $m \times r$ and Y size $n \times r$.

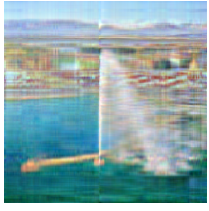
Many matrices are not low-rank, but can be approximated by low-rank matrices.

Example: We can see an image as a matrix, and do a low-rank approximation

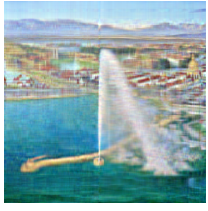
rank = 5



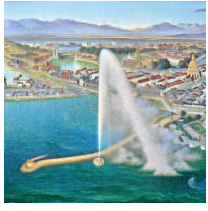
rank = 10



rank = 20



rank = 50



rank = 768

rank = 5

rank = 10

rank = 20

rank = 50

rank = 768

In general, a best low-rank approximation of A solves

$$\min_{B \text{ rank} \leq r} \|A - B\|$$

Solution: truncated SVD $A = U\Sigma V^\top$, then $B = U_r \Sigma_r V_r^\top$ where:

Matrix	Size	Property
U_r	$m \times r$	$U_r^\top U_r = I_r$
V_r	$n \times r$	$V_r^\top V_r = I_r$
Σ_r	$r \times r$	Diagonal

A rank $\leq r$ matrix uses $r(m+n) - r^2 < mn$ parameters. $\Rightarrow$ we can find a rank $\leq r$ approximation of A without knowing all the entries.

Optimization problem:

$$\min_{B \text{ rank } \leq r} \|\mathcal{P}_\Omega A - \mathcal{P}_\Omega B\|,$$

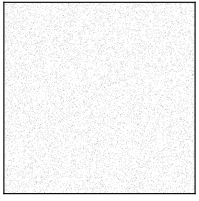
where $\mathcal{P}_\Omega$ projection onto known entries indexed by $\Omega \subset \{1, \ldots, m\} \times \{1 \ldots, n\}$.

Example: Image with 2.7% of points removed, restoration using rank 100.

Original

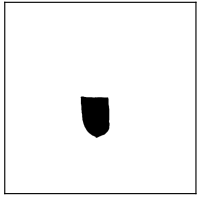
Reconstructed image #1

Mask #1



Reconstructed image #2

Mask #2

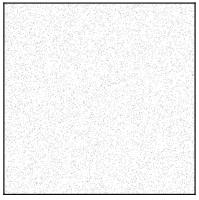


Example: Image with 2.7% of points removed, restoration using rank 100.

Original

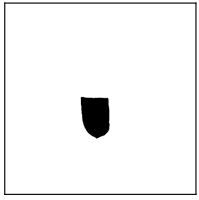
Reconstructed image #1

Mask #1



Reconstructed image #2

Mask #2

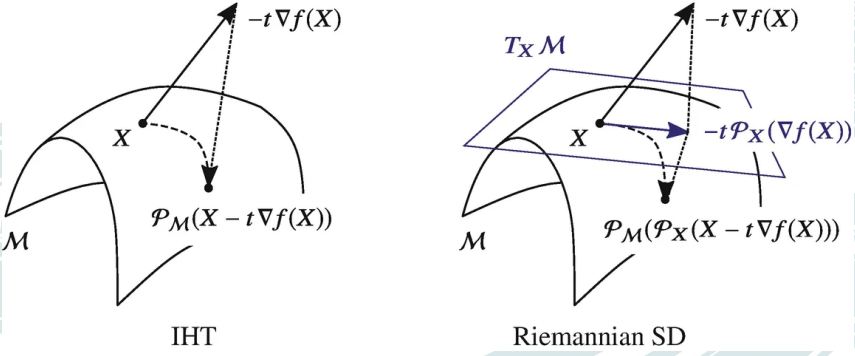


Techniques:

- Alternating least squares: Write $B = XY^\top$ and alternatingly optimize for X and Y.
- Convex optimization: Remove constraints and replace $\|\cdot\|$ by nuclear norm.
- Riemannian gradient descent: Constrained optimization on the manifold of rank $\leq r$ matrices.

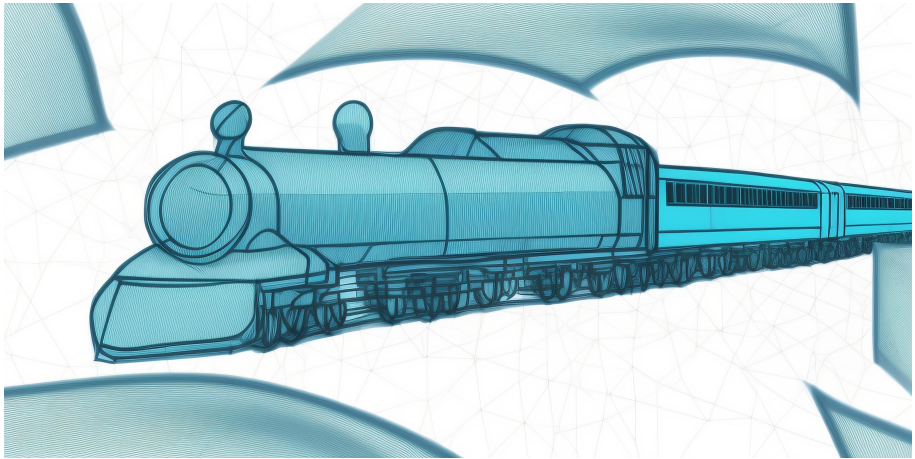
Manifold-constrained optimization, two approaches:

- Perform step in gradient direction → project down onto constrain set. Or:
- Project gradient onto tangent space → perform step in projected direction → project down onto constraint set.

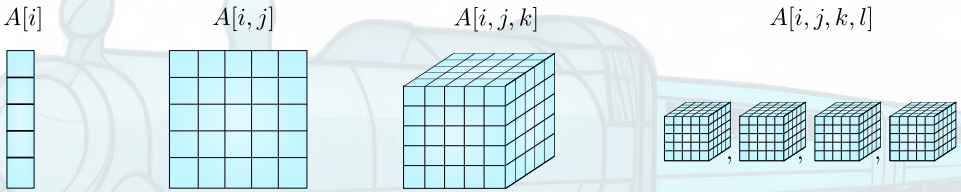


Credits: Vandereycken & Uschmajew

Part 2 | Tensors



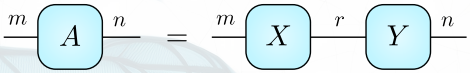
Tensors are high-dimensional arrays



	1D (vector)	2D (matrix)	3D	4D
Examples	Audio Stock prices	B&W images Excel spreadsheet	Color images B&W video Minecraft map MRI scan	Color video fMRI scan

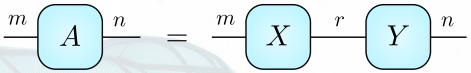
Low-rank matrix $\Leftrightarrow$ Product of two smaller matrices

$$A = XY^\top \;\rightarrow\; A[i,j] = \sum_{\ell=1}^{r} X[i,\ell]Y[j,\ell] \;\rightarrow$$



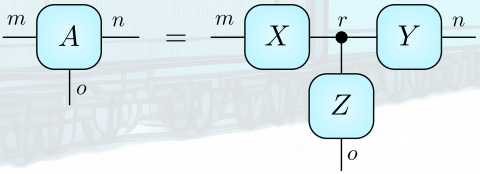
Low-rank matrix ⇔ Product of two smaller matrices

$$A = XY^\top \;\to\; A[i,j] = \sum_{\ell=1}^{r} X[i,\ell]Y[j,\ell] \;\to$$



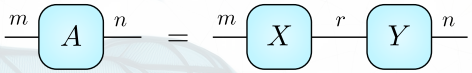
Low-rank tensor ⇔ Product of smaller tensors. **Example:**

$$A[i,j,k] = \sum_{\ell=1}^{r} X[i,\ell]Y[j,\ell]Z[k,\ell] \;\to$$



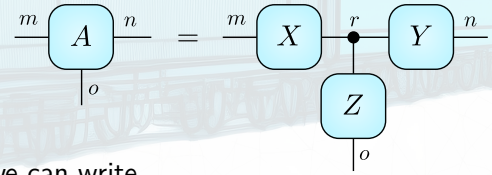
Low-rank matrix ⇔ Product of two smaller matrices

$$A = XY^\top \;\to\; A[i,j] = \sum_{\ell=1}^{r} X[i,\ell]Y[j,\ell] \;\to$$



Low-rank tensor ⇔ Product of smaller tensors. **Example:**

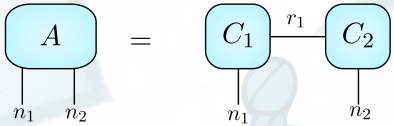
$$A[i,j,k] = \sum_{\ell=1}^{r} X[i,\ell]Y[j,\ell]Z[k,\ell] \;\to$$



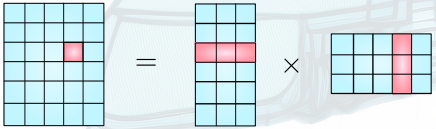
This is called a CP tensor. Equivalently we can write

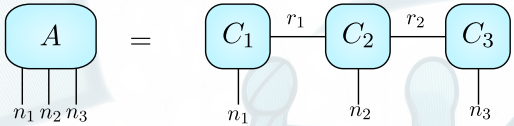
$$A = \sum_{\ell=1}^{r} x_\ell \otimes y_\ell \otimes z_\ell,$$

i.e. a sum of elementary tensors.

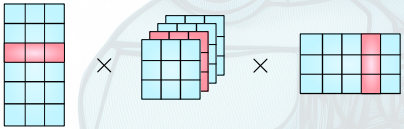


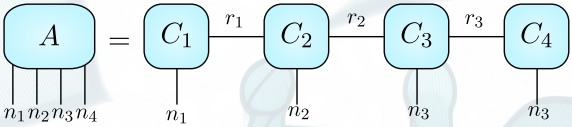
$$A[i_1, i_2] = \sum_{\ell_1=1}^{r_1} C_1[i_1, \ell_1] C_2[\ell_1, i_2]$$
$$= C_1[i_1, :] C_2[:, i_2]$$



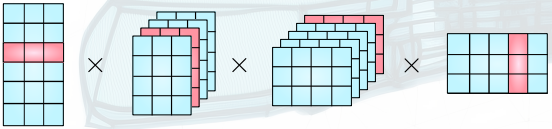


$$A[i_1, i_2, i_3] = C_1[i_1, :]C_2[:, i_2, :]C_3[:, i_3]$$
$$= \sum_{\ell_1=1}^{r_1} \sum_{\ell_2=1}^{r_2} C_1[i_1, \ell_1]C_2[\ell_1, i_2, \ell_2]C_3[\ell_2, i_3]$$



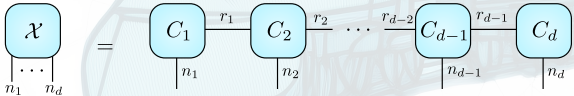


$$A[i_1, i_2, i_3, i_4] = C_1[i_1, :]C_2[:, i_2, :]C_3[:, i_3, :]C_4[:, i_4]$$
$$= \sum_{\ell_1=1}^{r_1} \sum_{\ell_2=1}^{r_2} \sum_{\ell_3=1}^{r_3} C_1[i_1, \ell_1]C_2[\ell_1, i_2, \ell_2]C_3[\ell_2, i_3, \ell_3]C_4[\ell_3, i_4]$$



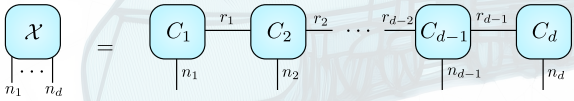
Advantages

- Easy to optimize
- Easy to manipulate and compute entries, dot products, etc.
- Scales well (curse of dimensionality)



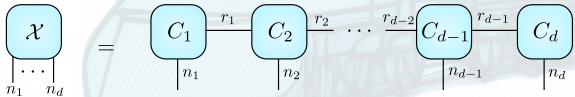
Advantages

- Easy to optimize
- Easy to manipulate and compute entries, dot products, etc.
- Scales well (curse of dimensionality)



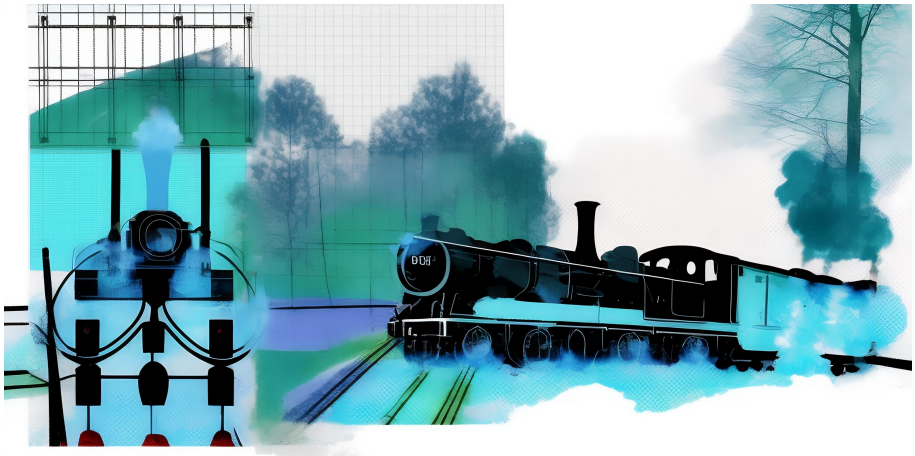
Advantages

- Easy to optimize
- Easy to manipulate and compute entries, dot products, etc.
- Scales well (curse of dimensionality)



Part 3 | TTML

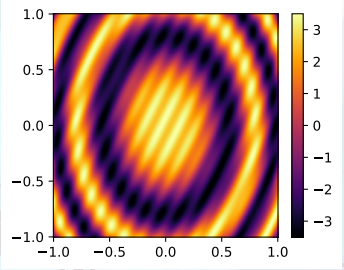
Joint work with Bart Vandereycken



§3 | Matrices as discretized functions

Many functions in 2D can be represented by low rank matrices.

$$f(x, y) = 3\cos(10(x^2 + y^2/2)) - \sin(20(2x - y))/2$$



$$X^\top =$$

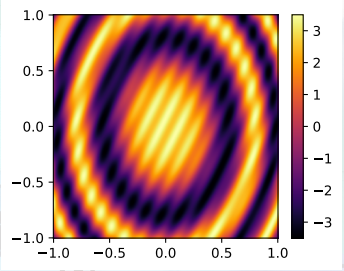
$$Y^\top =$$

Matrix $\Leftrightarrow$ function $I^2 \to \mathbb{R}$ discretized on a grid

§3 | Matrices as discretized functions

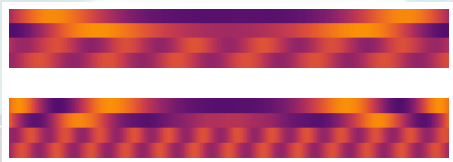
Many functions in 2D can be represented by low rank matrices.

$$f(x, y) = 3\cos(10(x^2 + y^2/2)) - \sin(20(2x - y))/2$$



$X^\top =$

$Y^\top =$



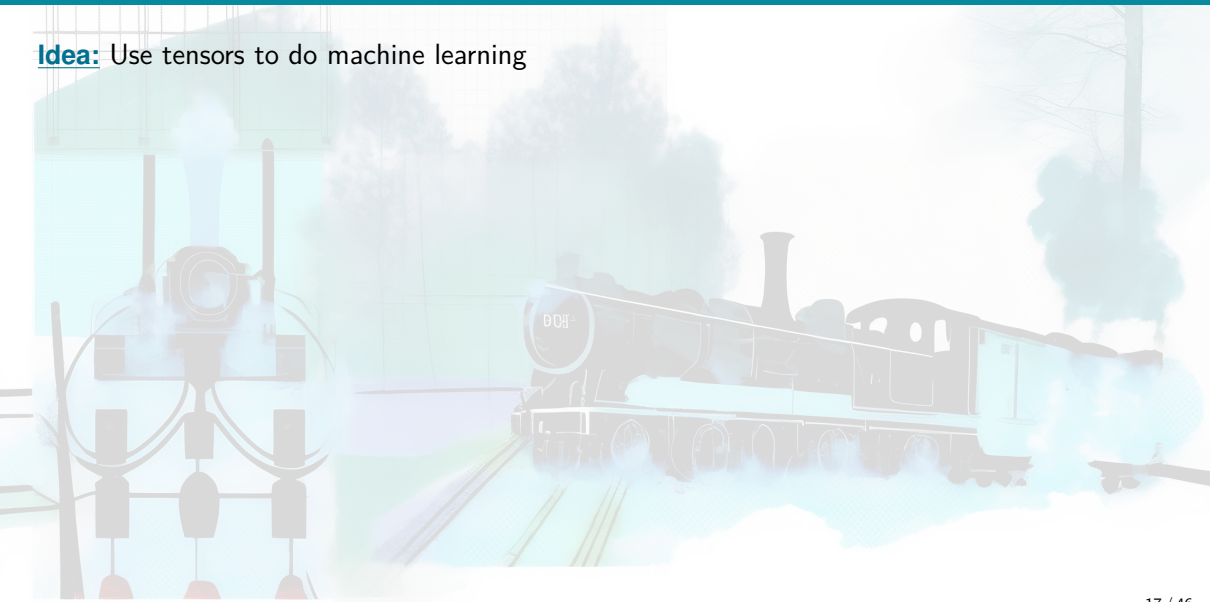
Matrix $\Leftrightarrow$ function $I^2 \to \mathbb{R}$ discretized on a grid

Tensor $\Leftrightarrow$ function $I^d \to \mathbb{R}$ discretized on a grid

- Low-rank tensors can represent complicated functions

§3 | Supervised learning

Idea: Use tensors to do machine learning



§3 | Supervised learning

Idea: Use tensors to do machine learning

Supervised learning. Learn a function $f\colon I^d \to \mathbb{R}$ based on a number of examples.

Example: Predict noise made by airplane wing in a wind tunnel.

Frequency (Hz)
Angle of attack ($\deg$)
Chord length (m) $\quad\to\quad$ Sound pressure (dB)
Wind tunnel airspeed (m/s)
Displacement thickness (m)

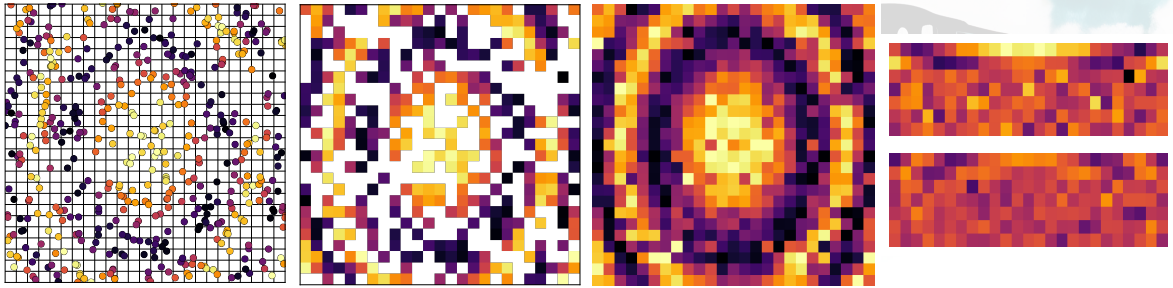
§3 | TTML: Matrix case

Matrix $A \Leftrightarrow$ function $\widehat{f}\colon I^2 \to \mathbb{R}$ discretized on a grid

Supervised learning. Learn a function $f\colon I^d \to \mathbb{R}$ based on a number of examples

Each value $\widehat{f}(x_1, x_2)$ corresponds to an entry A_{i_1,i_2} of a matrix.

- Learning $\widehat{f} \Leftrightarrow$ matrix completion of the matrix A.



§3 | TTML: Matrix case

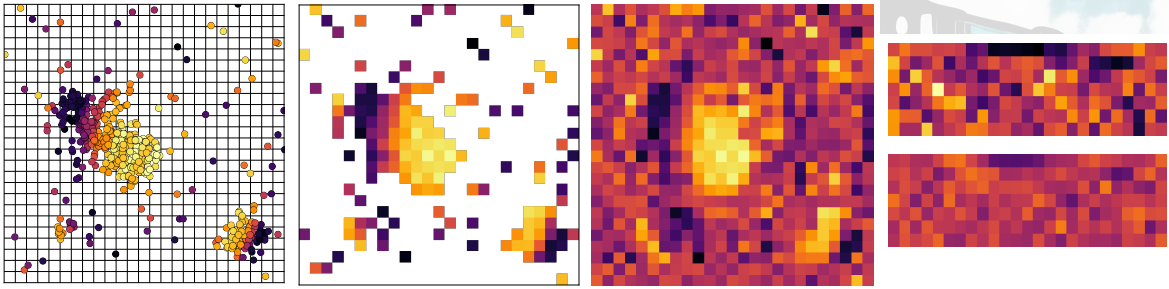
Matrix $A \Leftrightarrow$ function $\widehat{f}\colon I^2 \to \mathbb{R}$ discretized on a grid

Supervised learning. Learn a function $f\colon I^d \to \mathbb{R}$ based on a number of examples

Each value $\widehat{f}(x_1, x_2)$ corresponds to an entry A_{i_1,i_2} of a matrix.

- Learning $\widehat{f} \Leftrightarrow$ matrix completion of the matrix A.

Problem: Data is rarely uniformly distributed



§3 | TTML: Matrix case

Matrix $A \Leftrightarrow$ function $\widehat{f}\colon I^2 \to \mathbb{R}$ discretized on a grid

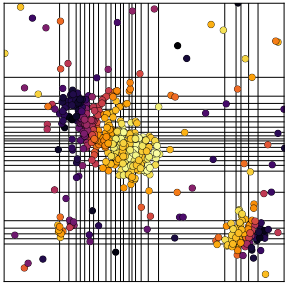
Supervised learning. Learn a function $f\colon I^d \to \mathbb{R}$ based on a number of examples

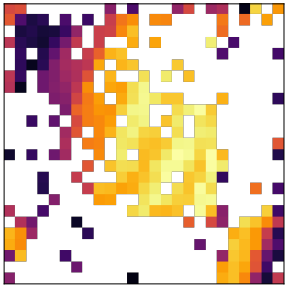
Each value $\widehat{f}(x_1, x_2)$ corresponds to an entry A_{i_1,i_2} of a matrix.

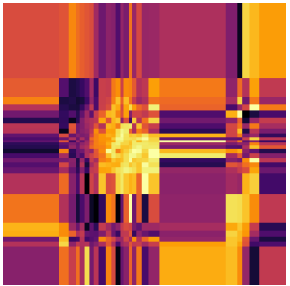
- Learning $\widehat{f} \Leftrightarrow$ matrix completion of the matrix A.

Problem: Data is rarely uniformly distributed

- Don't use a uniform grid





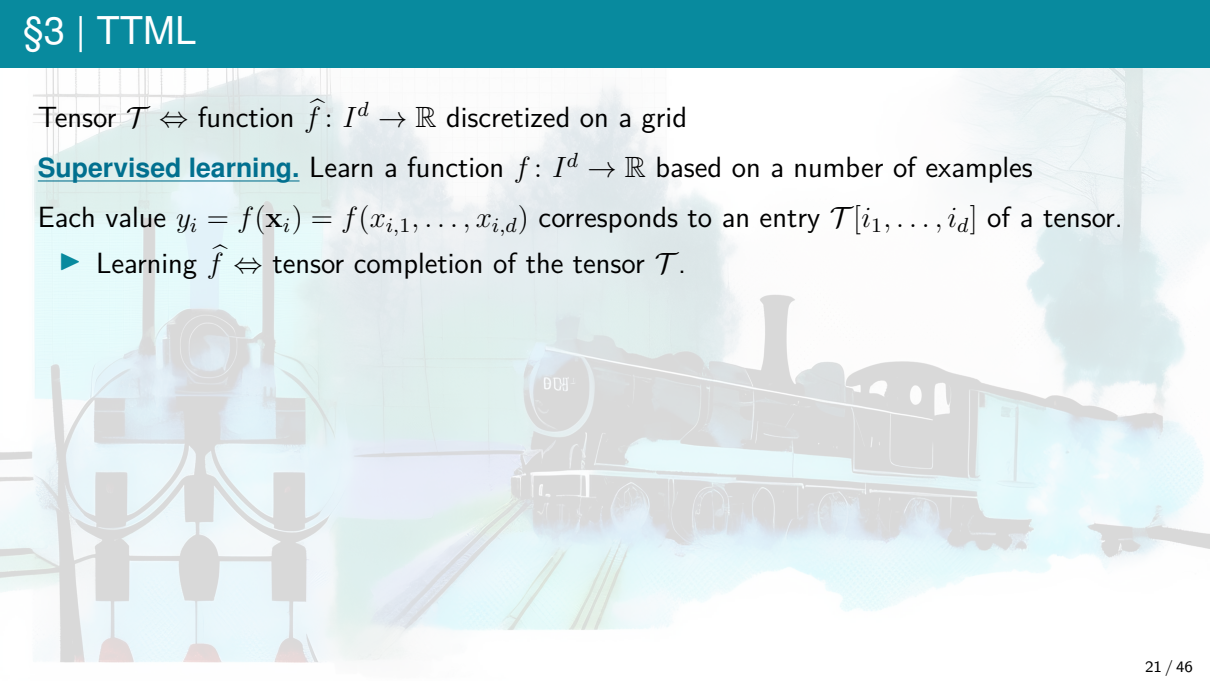


Tensor $\mathcal{T} \Leftrightarrow$ function $\widehat{f}\colon I^d \to \mathbb{R}$ discretized on a grid

Supervised learning. Learn a function $f\colon I^d \to \mathbb{R}$ based on a number of examples

Each value $y_i = f(\mathbf{x}_i) = f(x_{i,1}, \ldots, x_{i,d})$ corresponds to an entry $\mathcal{T}[i_1, \ldots, i_d]$ of a tensor.

- Learning $\widehat{f} \Leftrightarrow$ tensor completion of the tensor $\mathcal{T}$.



Tensor $\mathcal{T} \Leftrightarrow$ function $\widehat{f}\colon I^d \to \mathbb{R}$ discretized on a grid

Supervised learning. Learn a function $f\colon I^d \to \mathbb{R}$ based on a number of examples

Each value $y_i = f(\mathbf{x}_i) = f(x_{i,1}, \ldots, x_{i,d})$ corresponds to an entry $\mathcal{T}[i_1, \ldots, i_d]$ of a tensor.

- Learning $\widehat{f} \Leftrightarrow$ tensor completion of the tensor $\mathcal{T}$.

$$\min_{\mathcal{T} \in \mathcal{M}} \sum_{i=1}^{N} \left(\mathcal{T}[j_1(\mathbf{x}_i), \ldots, j_d(\mathbf{x}_i)] - y_i\right)^2, \qquad (\star)$$

$\mathcal{M}$ is set of all rank $\leq (r_1, \ldots, r_{d-1})$ tensor trains.

- Solving $(\star)$ using alternating least squares (ALS) leads to ill-conditioned subproblems for non-uniformly distributed data in practice.
- Riemannian gradient descent methods work well in practice

Tensor completion

$$\min_{\mathcal{T}\in\mathcal{M}} \sum_{i=1}^{N} \left(\mathcal{T}[j_1(\mathbf{x}_i), \ldots, j_d(\mathbf{x}_i)] - y_i\right)^2, \qquad (\star)$$

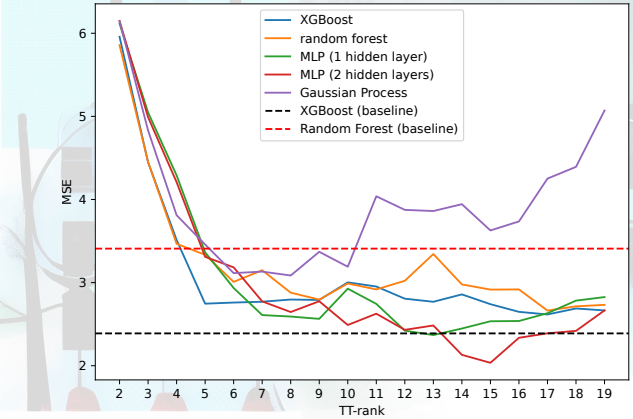
Iterative methods need good initialization.

Trick:

- First train a different ML model $g\colon \mathbb{R}^d \to \mathbb{R}$ (e.g. random forest, neural network) on the same training data.
- Discretize $g \Rightarrow$ obtain tensor $\mathcal{S}$.
- Use initialisation $\mathcal{T}_0$ such that $\mathcal{T}_0 \approx \mathcal{S}$.
- This is possible because we can sample arbitrary entries of $\mathcal{S}$. We use TT-cross.
- Quality of model obtained depends strongly on ML model g used for initialization.

Example: Predict noise made by airplane wing in a wind tunnel.

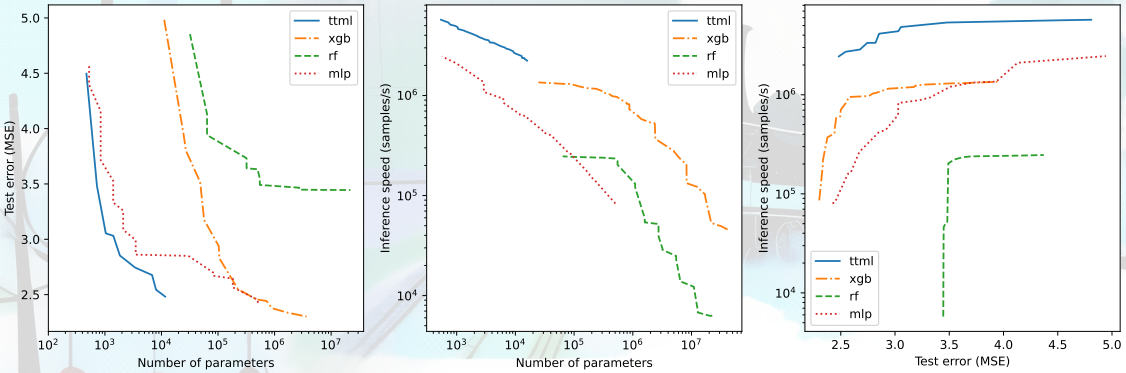
Frequency (Hz)
Angle of attack ($\deg$)
Chord length (m) $\rightarrow$ Sound pressure (dB)
Wind tunnel airspeed (m/s)
Displacement thickness (m)



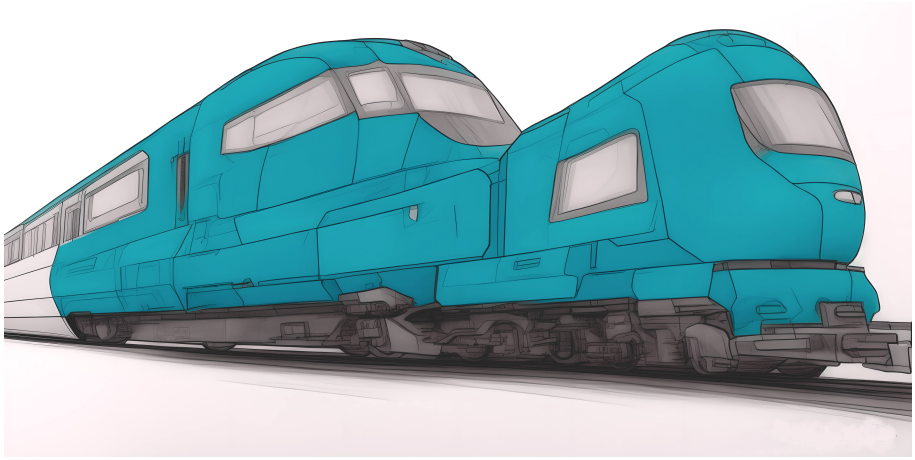
§3 | TTML: Experiments

For certain datasets, TTML is *faster*, uses fewer parameters/memory and gives better error than other methods.

- Memory and speed is usually excellent for TTML.
- Performance depends strongly on dataset used. TTML works best for .



Part 4 | Randomized linear algebra



Recall: Best rank r approximation of an $m \times n$ matrix is given by truncated SVD.

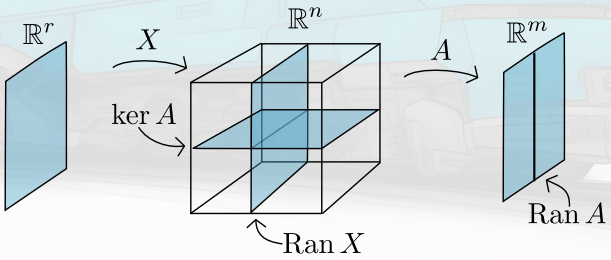
- **Problem:** SVD costs $O(mn^2)$ flops $(m > n)$.

Randomized matrix decomposition: First multiply A by a random $n \times r$ matrix X, then try to obtain truncated SVD from AX. We call X 'dimension reduction matrix' (DRM).

Lemma

If A is rank $\widehat{r} < r$ then AX has almost surely the same range as A.

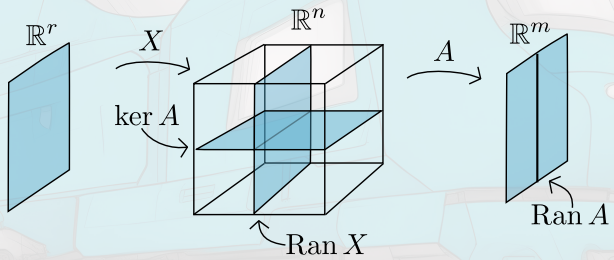
Proof: Counting dimensions.



Lemma

If A is rank $\widehat{r} < r$ then AX has almost surely the same range as A.

Proof: Counting dimensions.



Idea: If A can be well approximated by a rank-$\widehat{r}$ matrix $A_{\widehat{r}}$, then restricting the range to $\operatorname{Ran} AX$ keeps most of the information.

- That is, $\mathcal{P}_{AX} A \approx A$. More precisely $\|\mathcal{P}_{AX} A - A\| \leq C \|A_{\widehat{r}} - A\|$.
- Observe $\mathcal{P}_{AX} = QQ^\top$ where Q is an $m \times r$ orthonormal basis of $\operatorname{Ran} AX$.
- $Q(Q^\top A)$ is a rank r approximation.

Theorem (HMT)

Let X be $n \times r$ random normal with $r > \widehat{r} + 1$, and let $Q = \mathrm{Orth}(AX)$. Then

$$\mathbb{E}\,\|Q(Q^\top A) - A\|_F^2 \le \left(1 + \frac{\widehat{r}}{r - \widehat{r} - 1}\right) \|A_{\widehat{r}} - A\|_F^2$$

- Can compute good rank r approximation in $O(mnr) < O(mn^2)$ flops
- If A is a large structured (e.g. sparse) matrix, then computing $\mathrm{Orth}(AX)$ is expensive and difficult to parallelize

§4 | Generalized Nyström method

Idea: If m, n are both big, AX is still a big matrix.

- Use two DRMs and compute sketch $Y^\top AX$ where X is $n \times r^R$ and Y is $m \times r^L$.
- Obtain rank r^R approximation $AX(Y^\top AX)^\dagger Y^\top A$, where † denotes pseudoinverse.
- Can be written as *oblique* projection $\mathcal{P}_{AX,Y} A$.

Theorem (Tropp et al.)

Assume X, Y random normal and $r^L > r^R > \widehat{r} - 1$. The generalized Nyström method $A \approx \mathcal{P}_{AX,Y} A = AX(Y^\top AX)^\dagger Y^\top A$ then satisfies

$$\mathbb{E}\,\|\mathcal{P}_{AX,Y} A - A\|_F^2 \leq \left(1 + \frac{r^R}{r^L - r^R - 1}\right)\left(1 + \frac{\widehat{r}}{r^R - \widehat{r} - 1}\right)\|A_{\widehat{r}} - A\|_F^2$$

Theorem (Tropp et al.)

Assume X, Y *random normal and* $r^L > r^R > \widehat{r} - 1$. *The generalized Nyström method* $A \approx \mathcal{P}_{AX,Y} A = AX(Y^\top AX)^\dagger Y^\top A$ *then satisfies*

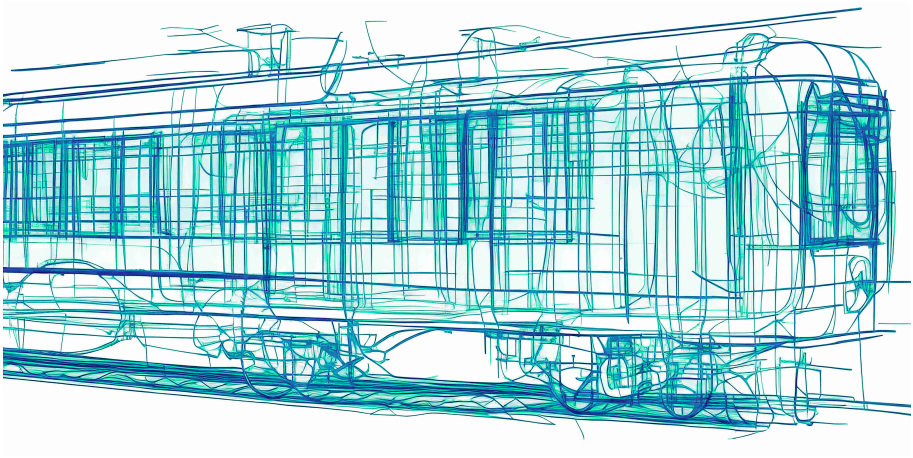
$$\mathbb{E}\,\|\mathcal{P}_{AX,Y} A - A\|_F^2 \leq \left(1 + \frac{r^R}{r^L - r^R - 1}\right)\left(1 + \frac{\widehat{r}}{r^R - \widehat{r} - 1}\right)\|A_{\widehat{r}} - A\|_F^2$$

Advantages

- Cheap for very large structured matrices
- Streaming method; we can compute an approximation of $A + B$ both in parallel while seeing each of the summands only once.

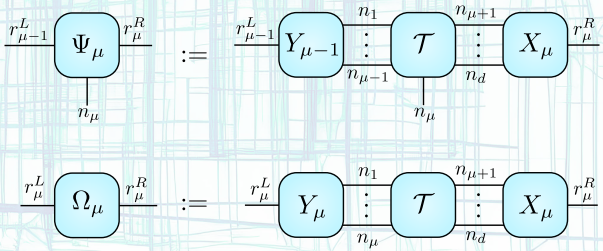
Part 5 | Randomized TT Approximations

Joint work with Bart Vandereycken and Daniel Kressner



Idea: Generalize the generalized Nyström method to tensor trains.

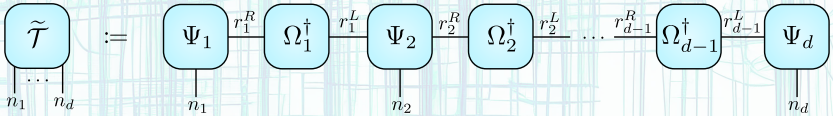
Step 1: Compute sketches – linear in the data and one-pass



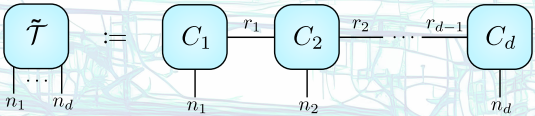
X_μ is a 'right' DRM of size $(n_{\mu+1} \cdots n_d) \times r^R_\mu$ for $1 \leq \mu \leq d-1$;

Y_μ is a 'left' DRM of size $(n_1 \cdots n_\mu) \times r^L_\mu$ for $1 \leq \mu \leq d-1$.

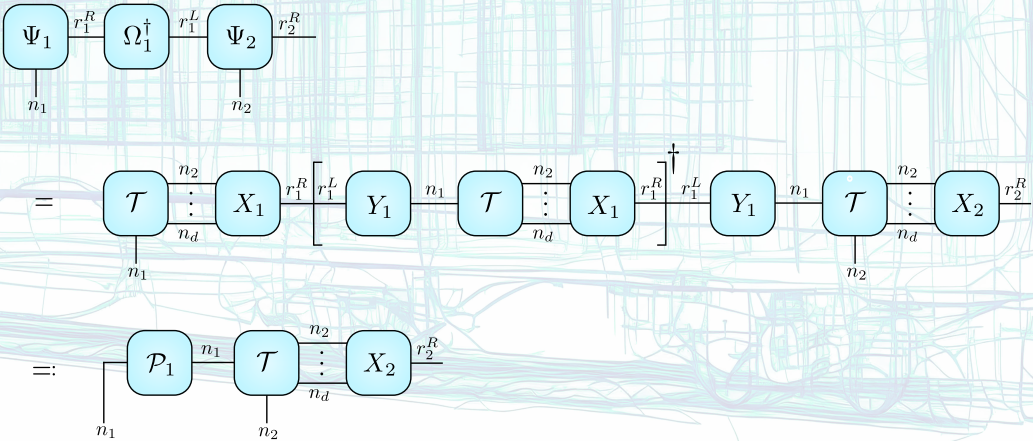
Step 2: Assemble in a tensor network – zero cost



Step 3: Reduce to standard TT – optional and cheap

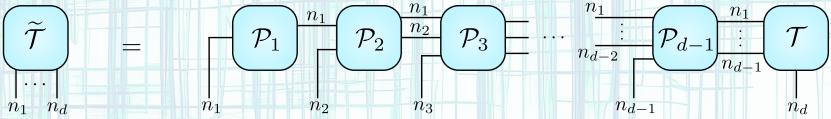


- Write approximation as projector



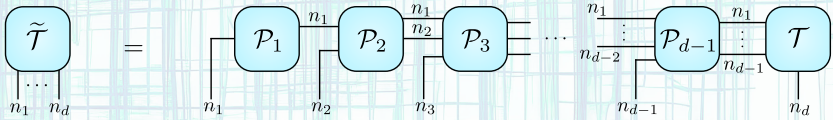
§5 | Why does STTA work?

- Write approximation as projector



§5 | Why does STTA work?

- Write approximation as projector



- Each projector is **oblique**
- For TT-SVD and TT-HMT approximation has **orthogonal** projectors
- Effect of each projector on error can be analyzed independently

§5 | Error bound in expectation

Theorem

Let $\widetilde{\mathcal{T}}$ be obtained using STTA, and let $\mathcal{T}_{\widehat{r}}$ be a best rank $\widehat{\mathbf{r}} = (\widehat{r}_1, \ldots, \widehat{r}_{d-1})$ TT-approximation with $\widehat{r}_\mu + 1 < r_\mu^R < r_\mu^L - 1$. Then

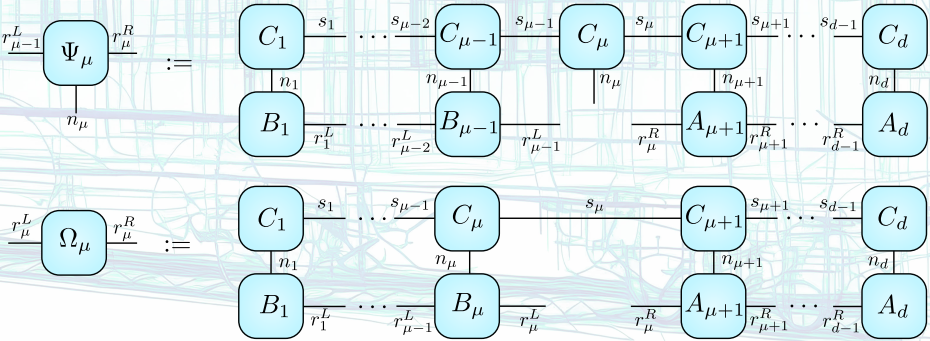
$$\mathbb{E}\,\|\widetilde{\mathcal{T}} - \mathcal{T}\|_F \leq \left(\sum_{\mu=1}^{d-1} \left[\prod_{\alpha=1}^{\mu-1} c_\alpha \right] c'_\mu \right) \|\mathcal{T}_{\widehat{\mathbf{r}}} - \mathcal{T}\|_F,$$

with

$$c_\mu = 1 + \sqrt{\frac{r_\mu^R}{r_\mu^L - r_\mu^R - 1}}, \qquad c'_\mu = \sqrt{1 + \frac{r_\mu^R}{r_\mu^L - r_\mu^R - 1}} \cdot \sqrt{1 + \frac{\widehat{r}_\mu}{r_\mu^R - \widehat{r}_\mu - 1}}.$$

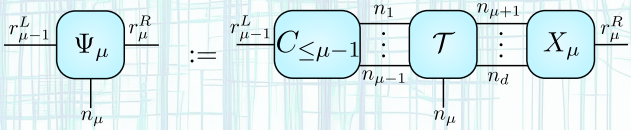
- Like GN and HMT, error is within a constant of optimal error.
- **Practical performance** is better than theoretical; no exponential dependence on d.
- Very paralellizable and streamable

- Using TTs as DRMs we can do **fast structured sketches**.
- Efficient methods for CP, Tucker and sparse tensors, and any linear combination of these (e.g., TT + sparse).
- Example: diagram for computing Ψ_μ, Ω_μ when $\mathcal{T}$ is a TT.



§5 | Orthogonalized Tensor Train Sketch (OTTS)

- We can modify STTA to improve accuracy at the cost of speed and streamability.
- Just replace left DRM in definition of Ψ_μ with previous cores, and do orthogonalization.

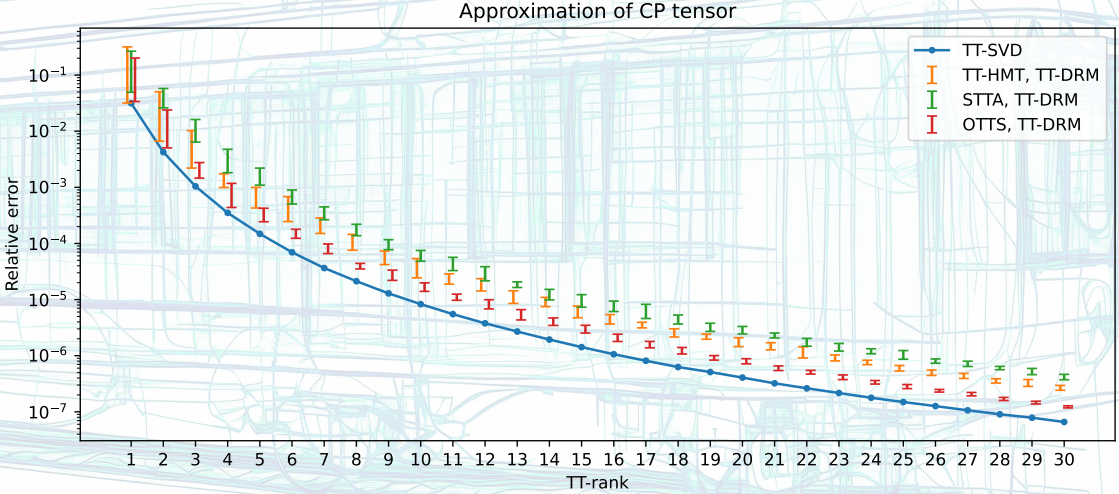


Theorem

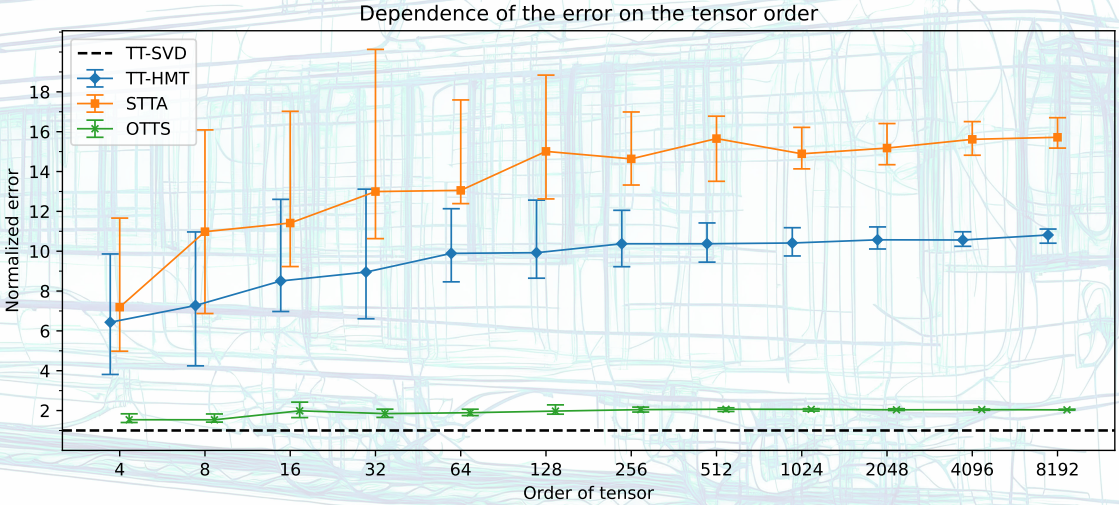
Let $\widetilde{\mathcal{T}}$ be obtained using OTTS, and let $\mathcal{T}_{\widehat{\mathbf{r}}}$ be a best rank $\widehat{\mathbf{r}} = (\widehat{r}_1, \ldots, \widehat{r}_{d-1})$ TT-approximation with $\widehat{r}_\mu + 1 < r^R_\mu < r^L_\mu - 1$. Then

$$\mathbb{E}\,\|\widetilde{\mathcal{T}} - \mathcal{T}\|_F^2 \left(\sum_{\mu=1}^{d-1} (c'_\mu)^2 \right) \|\mathcal{T}_{\widehat{\mathbf{r}}} - \mathcal{T}\|_F^2,$$

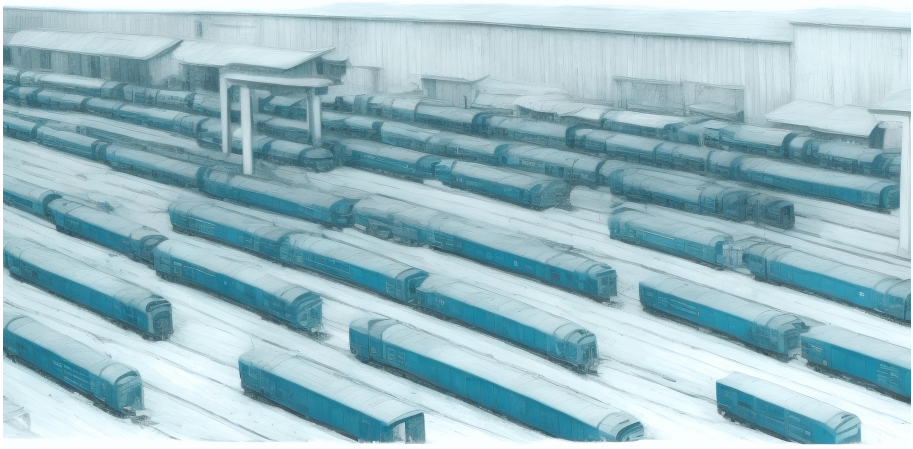
§5 | Experiments



§5 | Experiments



Part 6 | Sketched low-rank solvers



§6 | Low-rank linear systems

Consider the generalized Sylvester equation:

$$\mathcal{L}(H) = AH + HB + \sum_{i=1}^{\ell} M_i H N_i = C$$

We want to find low-rank solutions.

- Need low-rank approximation of $\mathcal{L}(H)$
- Low-rank streaming methods like GN and STTA are excellent for this

Consider the generalized Sylvester equation:

$$\mathcal{L}(H) = AH + HB + \sum_{i=1}^{\ell} M_i H N_i = C$$

We want to find low-rank solutions.

- Need low-rank approximation of $\mathcal{L}(H)$
- Low-rank streaming methods like GN and STTA are excellent for this

Idea: Use properties of low-rank streaming methods to develop general approaches for solving low-rank linear systems.

Example: Generalized Nyström

We have an approximation $\mathcal{S}_{X,Y}(A) = AX(Y^\top AX)^\dagger Y^\top A$.

- Computing inner product $\langle A, B\rangle$ of two rank-r matrices costs $O(r^2(m+n))$ flops.
- Approximation $\langle A, B\rangle_{X,Y} = \langle Y^\top AX, Y^\top BX\rangle$ costs only $O(r^2)$.

Example: Generalized Nyström

We have an approximation $\mathcal{S}_{X,Y}(A) = AX(Y^\top AX)^\dagger Y^\top A$.

- Computing inner product $\langle A, B\rangle$ of two rank-r matrices costs $O(r^2(m+n))$ flops.
- Approximation $\langle A, B\rangle_{X,Y} = \langle Y^\top AX, Y^\top BX\rangle$ costs only $O(r^2)$.

Why does this work?

Compatibility with the sketched approximation $\mathcal{S}_{X,Y}$:

- $\langle A, B\rangle_{X,Y} = \langle A, \mathcal{S}_{X,Y}(B)\rangle_{X,Y} = \langle \mathcal{S}_{X,Y}(A), \mathcal{S}_{X,Y}(B)\rangle_{X,Y}$.
- $\langle A, A\rangle_{X,Y} = 0 \quad \Leftrightarrow \quad \mathcal{S}_{X,Y}(A) = 0$.

§6 | Sketched low-rank solvers

We formalize the notions of *linear low-rank streaming method* and with a compatible bilinear form to get:

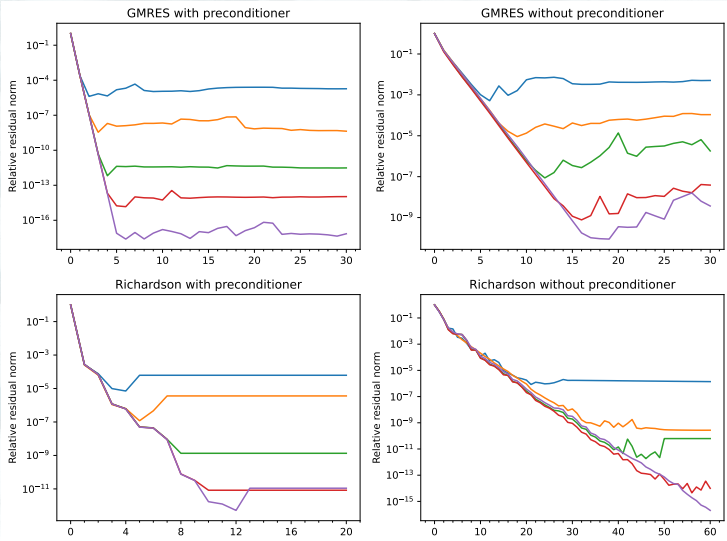
- Sketched version of Richardson iteration.
- Sketched version of Krylov methods (GMRES).

These methods are:

- generalist and apply to any linear system;
- compatible with preconditioning;
- linear in the number of terms in linear operator $\mathcal{L}$;
- generalize existing sketched methods using e.g. $\mathcal{S}_{X,Y}(A) = \mathcal{P}_Y A \mathcal{P}_X$.

Does not work for STTA yet; no obvious compatible bilinear form.

§6 | Experiment



$$\mathcal{L}(H) = AH + HA^\top + \gamma^2 \left(N_1 H N_1^\top + N_2 H N_2^\top \right) = C,$$

A, N_i tridiagonal, C rank 2.

Preconditioner: Lyapunov solver.

Part 7 | The part where you're supposed to ask questions