Approximation to multivariate functions in the extended tensor train format

Bonan Sun

EPF Lausanne, Switzerland

bonan.sun@epfl.ch

https://bonans.github.io/

Based on a joint work with
Christoph Strössner and Daniel Kressner

SIAM LA24, May 14, 2024

Organization

1. Functional low-rank approximation
2. The extended functional tensor train (EFTT) format
3. EFTT approximation algorithm
4. Numerical experiments

Organization

1. Functional low-rank approximation
2. The extended functional tensor train (EFTT) format
3. EFTT approximation algorithm
4. Numerical experiments

Functional low-rank approximations

- Consider $f : [-1,1]^d \to \mathbb{R}$. Simplest low-rank structure of f is the **separation of variables**:

$$f(x_1, x_2, \cdots, x_d) \approx \sum_{\alpha=1}^{R} g_\alpha^{(1)}(x_1) g_\alpha^{(2)}(x_2) \cdots g_\alpha^{(d)}(x_d),$$

which uses dR **univariate functions** $g_\alpha^{(i)} : [-1,1] \to \mathbb{R}$.

Functional low-rank approximations

- Consider $f : [-1,1]^d \to \mathbb{R}$. Simplest low-rank structure of f is the **separation of variables**:

$$f(x_1, x_2, \cdots, x_d) \approx \sum_{\alpha=1}^{R} g_\alpha^{(1)}(x_1) g_\alpha^{(2)}(x_2) \cdots g_\alpha^{(d)}(x_d),$$

  which uses dR **univariate functions** $g_\alpha^{(i)} : [-1,1] \to \mathbb{R}$.
- Often beneficial to impose additional low-rank structures.

Functional low-rank approximations

- Consider $f : [-1,1]^d \to \mathbb{R}$. Simplest low-rank structure of f is the **separation of variables**:

$$f(x_1, x_2, \cdots, x_d) \approx \sum_{\alpha=1}^{R} g_\alpha^{(1)}(x_1) g_\alpha^{(2)}(x_2) \cdots g_\alpha^{(d)}(x_d),$$

  which uses dR **univariate functions** $g_\alpha^{(i)} : [-1,1] \to \mathbb{R}$.
- Often beneficial to impose additional low-rank structures.
- This work focuses on the **functional tensor train** (FTT) structure $\rightsquigarrow$

$$f(x_1, \ldots, x_d) \approx \sum_{\alpha_1=1}^{R_1} \ldots, \sum_{\alpha_{d-1}=1}^{R_{d-1}} g_{1,\alpha_1}^{(1)}(x_1) g_{\alpha_1,\alpha_2}^{(2)}(x_2) \cdots g_{\alpha_{d-2},\alpha_{d-1}}^{(d-1)}(x_{d-1}) g_{\alpha_{d-1},1}^{(d)}(x_d),$$

  which uses dR^2 **univariate functions** $g_{\alpha_{i-1},\alpha_i}^{(i)} : [-1,1] \to \mathbb{R}$.

Functional low-rank approximations

- For $f : [-1,1]^d \to \mathbb{R}$, functional tensor train format:

$$f(x_1,\dots,x_d) \approx \sum_{\alpha_1=1}^{R_1} \dots, \sum_{\alpha_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,\alpha_1}(x_1) g^{(2)}_{\alpha_1,\alpha_2}(x_2) \cdots g^{(d-1)}_{\alpha_{d-2},\alpha_{d-1}}(x_{d-1}) g^{(d)}_{\alpha_{d-1},1}(x_d)$$

$$= \begin{bmatrix} g^{(1)}_{1,1}(x_1) & \cdots & g^{(1)}_{1,R_1}(x_1) \end{bmatrix} \begin{bmatrix} g^{(2)}_{1,1}(x_2) & \cdots & g^{(2)}_{1,R_2}(x_2) \\ \vdots & \ddots & \vdots \\ g^{(2)}_{R_1,1}(x_2) & \cdots & g^{(2)}_{R_1,R_2}(x_2) \end{bmatrix} \cdots \begin{bmatrix} g^{(d-1)}_{1,1}(x_{d-1}) & \cdots & g^{(d-1)}_{1,R_{d-1}}(x_{d-1}) \\ \vdots & \ddots & \vdots \\ g^{(d-1)}_{R_{d-2},1}(x_{d-1}) & \cdots & g^{(d-1)}_{R_{d-2},R_{d-1}}(x_{d-1}) \end{bmatrix} \begin{bmatrix} g^{(d)}_{1,1}(x_d) \\ \vdots \\ g^{(d)}_{R_{d-1},1}(x_d) \end{bmatrix}$$

Functional low-rank approximations

- For $f : [-1,1]^d \to \mathbb{R}$, functional tensor train format:

$$f(x_1, \ldots, x_d) \approx$$

$$\begin{bmatrix} g^{(1)}_{1,1}(x_1) & \cdots & g^{(1)}_{1,R_1}(x_1) \end{bmatrix} \begin{bmatrix} g^{(2)}_{1,1}(x_2) & \cdots & g^{(2)}_{1,R_2}(x_2) \\ \vdots & \ddots & \vdots \\ g^{(2)}_{R_1,1}(x_2) & \cdots & g^{(2)}_{R_1,R_2}(x_2) \end{bmatrix} \cdots \begin{bmatrix} g^{(d-1)}_{1,1}(x_{d-1}) & \cdots & g^{(d-1)}_{1,R_{d-1}}(x_{d-1}) \\ \vdots & \ddots & \vdots \\ g^{(d-1)}_{R_{d-2},1}(x_{d-1}) & \cdots & g^{(d-1)}_{R_{d-2},R_{d-1}}(x_{d-1}) \end{bmatrix} \begin{bmatrix} g^{(d)}_{1,1}(x_d) \\ \vdots \\ g^{(d)}_{R_{d-1},1}(x_d) \end{bmatrix}$$

- For $\mathcal{T} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, (discrete) tensor train format:

$$\mathcal{T}_{i_1,\cdots,i_d} \approx$$

$$\begin{bmatrix} \mathcal{G}^{(1)}_{1,i_1,1} & \cdots & \mathcal{G}^{(1)}_{1,i_1,R_1} \end{bmatrix} \begin{bmatrix} \mathcal{G}^{(2)}_{1,i_2,1} & \cdots & \mathcal{G}^{(2)}_{1,i_2,R_2} \\ \vdots & \ddots & \vdots \\ \mathcal{G}^{(2)}_{R_1,i_2,1} & \cdots & \mathcal{G}^{(2)}_{R_1,i_2,R_2} \end{bmatrix} \cdots \begin{bmatrix} \mathcal{G}^{(d-1)}_{1,i_{d-1},1} & \cdots & \mathcal{G}^{(d-1)}_{1,i_{d-1},R_{d-1}} \\ \vdots & \ddots & \vdots \\ \mathcal{G}^{(d-1)}_{R_{d-2},i_{d-1},1} & \cdots & \mathcal{G}^{(d-1)}_{R_{d-2},i_{d-1},R_{d-1}} \end{bmatrix} \begin{bmatrix} \mathcal{G}^{(d)}_{1,i_d,1} \\ \vdots \\ \mathcal{G}^{(d)}_{R_{d-1},i_d,1} \end{bmatrix}$$

Functional low-rank approximation: applications

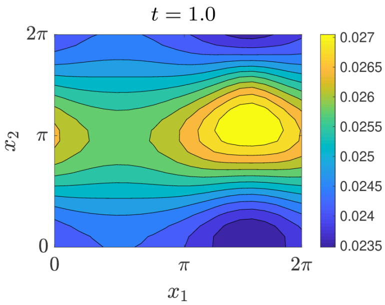
FTT format:

$$f(x_1,\dots,x_d) \approx \sum_{\alpha_1=1}^{R_1}\dots,\sum_{\alpha_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,\alpha_1}(x_1)g^{(2)}_{\alpha_1,\alpha_2}(x_2)\cdots g^{(d)}_{\alpha_{d-1},1}(x_d)$$

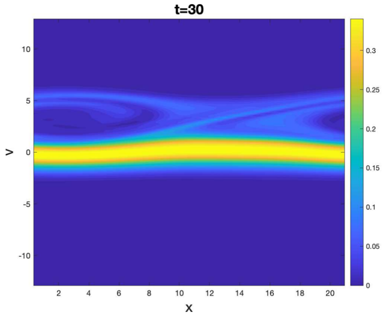
- Various operations can be performed efficiently, e.g., integration:

$$\int_{[-1,1]^d} f(x_1,\dots,x_d)\,\mathrm{d}\,x_1\cdots\mathrm{d}\,x_d \approx$$
$$\sum_{\alpha_1=1}^{R_1}\dots,\sum_{\alpha_{d-1}=1}^{R_{d-1}}\int_{-1}^{1} g^{(1)}_{1,\alpha_1}(x_1)\,\mathrm{d}\,x_1\int_{-1}^{1} g^{(2)}_{\alpha_1,\alpha_2}(x_2)\,\mathrm{d}\,x_2\cdots\int_{-1}^{1} g^{(d)}_{\alpha_{d-1},1}(x_d)\,\mathrm{d}\,x_d.$$

Functional low-rank approximation: applications



(a) FTT for Fokker-Planck equation [Dektor, Rodgers, Venturi 2021]



(b) FHT for Vlasov-Poisson equation [Guo, Qiu 2024]

- Applications: solution of PDEs [Bachmayr, Schneider, Uschmajew 2016], data compression [Rai, Kolla, Cannada, Gorodetsky 2019], optimal control [Oster, Sallandt, Schneider 2016], UQ [Dolgov, Scheichl 2019], etc.

Organization

Black-box FTT approximation

For $f : [-1,1]^d \to \mathbb{R}$, its FTT format reads:

$$f(x_1,\ldots,x_d) \approx \sum_{\alpha_1=1}^{R_1} \ldots, \sum_{\alpha_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,\alpha_1}(x_1) g^{(2)}_{\alpha_1,\alpha_2}(x_2) \cdots g^{(d-1)}_{\alpha_{d-2},\alpha_{d-1}}(x_{d-1}) g^{(d)}_{\alpha_{d-1},1}(x_d).$$

- Goal: given a black box function f, construct its FTT approximation using **as few function evaluations as possible**.
- Previous work — two different routes:
  1. Continuous version of TT-cross [Gorodetsky, Karaman, Marzouk 2019]
  2. Multivariate interpolation (basis expansion) + discrete TT

     [Haberstich, Nouy, Perrin 2023], [Bigoni, Engsig-Karup, Marzouk 2016], Chebfun [Trefethen et al. from 2004]
- We follow the second route in this work, but with an additional low-rank structure.

FTT via tensorized Chebyshev interpolation + TT

The second route: multivariate interpolation + discrete TT.

- Multivariate **Chebyshev interpolant** $\tilde{f}$ of f is given by

$$f(x_1, \ldots, x_d) \approx \tilde{f}(x_1, \ldots, x_d) = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \mathcal{A}_{i_1,\ldots,i_d} T_{i_1}(x_1) \cdots T_{i_d}(x_d).$$

$\mathcal{A}$ contains Chebyshev coefficients, computed by

$\mathcal{A} = \mathcal{T} \times_1 F^{(1)} \times_2 F^{(2)} \times_3 \cdots \times_d F^{(d)}$, for some DCT matrices $F^{(\ell)} \in \mathbb{R}^{n_\ell \times n_\ell}$

$\mathcal{T}_{i_1,\ldots,i_d} = f\left(x_{i_1}^{(1)}, \ldots, x_{i_d}^{(d)}\right)$, $x_k^{(\ell)} = \cos(\pi k/(n_\ell - 1))$, $k = 1, \ldots, n_\ell$, $\ell = 1, \ldots, d$.

FTT via tensorized Chebyshev interpolation + TT

The second route: multivariate interpolation + discrete TT.

- Multivariate **Chebyshev interpolant** $\tilde{f}$ of f is given by

$$f(x_1,\ldots,x_d) \approx \tilde{f}(x_1,\ldots,x_d) = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \mathcal{A}_{i_1,\ldots,i_d} T_{i_1}(x_1)\cdots T_{i_d}(x_d).$$

  $\mathcal{A}$ contains Chebyshev coefficients, computed by

$$\mathcal{A} = \mathcal{T} \times_1 F^{(1)} \times_2 F^{(2)} \times_3 \cdots \times_d F^{(d)}, \text{ for some DCT matrices } F^{(\ell)} \in \mathbb{R}^{n_\ell \times n_\ell}$$

$$\mathcal{T}_{i_1,\ldots,i_d} = f\left(x_{i_1}^{(1)},\ldots,x_{i_d}^{(d)}\right),\ x_k^{(\ell)} = \cos(\pi k/(n_\ell - 1)),\ k = 1,\ldots,n_\ell,\ \ell = 1,\ldots,d.$$

- Claim: Replacing $\mathcal{T}$ by a TT approximation $\hat{\mathcal{T}}$ yields an FTT approximation $\hat{f}$.

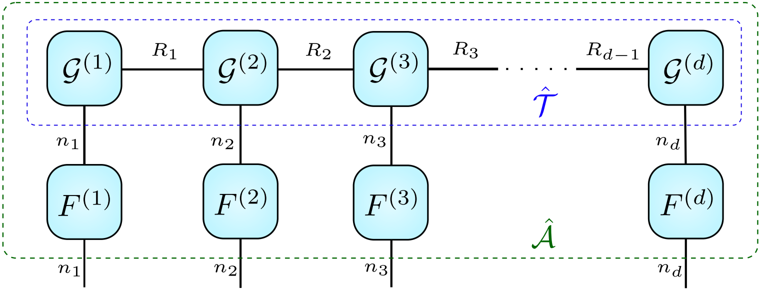
FTT via tensorized Chebyshev interpolation + TT

The second route: multivariate interpolation + discrete TT.

- Multivariate **Chebyshev interpolant** $\tilde{f}$ of f is given by

$$\tilde{f}(x_1,\dots,x_d) = \sum_{i_1=1}^{n_1}\cdots\sum_{i_d=1}^{n_d} \mathcal{A}_{i_1,\dots,i_d} T_{i_1}(x_1)\cdots T_{i_d}(x_d), \ \mathcal{A} = \mathcal{T}\times_1 F^{(1)}\times_2\cdots\times_d F^{(d)}$$

- Claim: Replacing $\mathcal{T}$ by a TT approximation $\hat{\mathcal{T}}$ yields an FTT approximation $\hat{f}$.



FTT via tensorized Chebyshev interpolation + TT

The second route: multivariate interpolation + discrete TT.

- Multivariate **Chebyshev interpolant** $\tilde{f}$ of f is given by

$$\tilde{f}(x_1,\ldots,x_d) = \sum_{i_1=1}^{n_1}\cdots\sum_{i_d=1}^{n_d} \mathcal{A}_{i_1,\ldots,i_d} T_{i_1}(x_1)\cdots T_{i_d}(x_d), \;\; \mathcal{A} = \mathcal{T}\times_1 F^{(1)} \times_2 \cdots \times_d F^{(d)}$$

- Claim: Replacing $\mathcal{T}$ by a TT approximation $\hat{\mathcal{T}}$ yields an FTT approximation $\hat{f}$:

$$\hat{\mathcal{T}}_{i_1,\ldots,i_d} = \sum_{\alpha_1=1}^{R_1}\cdots\sum_{\alpha_{d-1}=1}^{R_{d-1}} \mathcal{G}^{(1)}_{1,i_1,\alpha_1}\mathcal{G}^{(2)}_{\alpha_1,i_1,\alpha_2}\cdots\mathcal{G}^{(d-1)}_{\alpha_{d-2},i_{d-1},\alpha_{d-1}}\mathcal{G}^{(d)}_{\alpha_{d-1},i_d,1},$$

$$\Longrightarrow\; f \approx \hat{f} = \sum_{\alpha_1=1}^{R_1}\ldots,\sum_{\alpha_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,\alpha_1}(x_1) g^{(2)}_{\alpha_1,\alpha_2}(x_2)\cdots g^{(d-1)}_{\alpha_{d-2},\alpha_{d-1}}(x_{d-1}) g^{(d)}_{\alpha_{d-1},1}(x_d)$$

where $g^{(\ell)}_{\alpha_{\ell-1},\alpha_\ell}(x_\ell) = \sum_{j=1}^{n_\ell}\sum_{k=1}^{n_\ell} F^{(\ell)}_{j,k}\mathcal{G}^{(\ell)}_{\alpha_{\ell-1},k,\alpha_\ell} T_j(x_\ell) = \sum_{k=1}^{n_\ell}\left(\mathcal{G}^{(\ell)}\times_2 F^{(\ell)}\right)_{\alpha_{\ell-1},k,\alpha_\ell} T_k(x_\ell).$

FTT via tensorized Chebyshev interpolation + TT

The second route: multivariate interpolation + discrete TT.

- Multivariate **Chebyshev interpolant** $\tilde{f}$ of f is given by

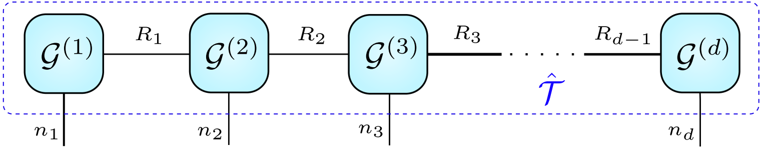
$$\tilde{f}(x_1, \ldots, x_d) = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \mathcal{A}_{i_1,\ldots,i_d} T_{i_1}(x_1) \cdots T_{i_d}(x_d), \ \mathcal{A} = \mathcal{T} \times_1 F^{(1)} \times_2 \cdots \times_d F^{(d)}$$

- Claim: Replacing $\mathcal{T}$ by a TT approximation $\hat{\mathcal{T}}$ yields an FTT approximation $\hat{f}$.
- Update Goal: Approximate $\mathcal{T}$ in TT format using **as few function evaluations as possible**.

Black box TT approximation

- New Goal: Approximate $\mathcal{T}$ in TT format using **as few function evaluations as possible** where

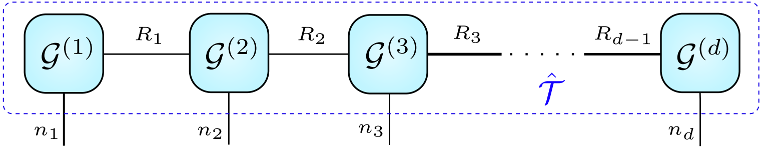
$$\mathcal{T}_{i_1,\ldots,i_d} = f(x^{(1)}_{i_1}, \ldots, x^{(d)}_{i_d}).$$



Black box TT approximation

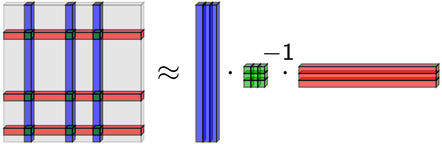
- New Goal: Approximate $\mathcal{T}$ in TT format using **as few function evaluations as possible** where

$$\mathcal{T}_{i_1,\ldots,i_d} = f(x_{i_1}^{(1)}, \ldots, x_{i_d}^{(d)}).$$



- Previous work: based on the Adaptive Cross Approximation (ACA) for matrices.

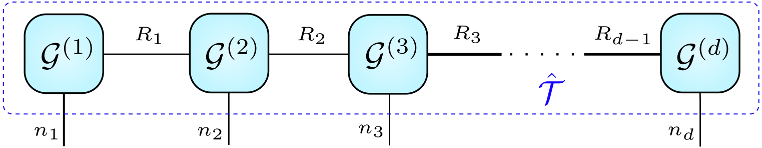
$$M \approx M(:, \mathcal{J}) M(\mathcal{I}, \mathcal{J})^{-1} M(\mathcal{I}, :) \rightsquigarrow$$



Black box TT approximation

- New Goal: Approximate $\mathcal{T}$ in TT format using **as few function evaluations as possible** where

$$\mathcal{T}_{i_1,\ldots,i_d} = f(x^{(1)}_{i_1}, \ldots, x^{(d)}_{i_d}).$$

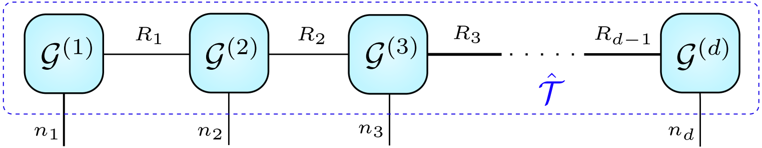


- Previous work: based on the Adaptive Cross Approximation (ACA) for matrices.
- Many variants of ACA for TT: `TT-cross`[Oseledets & Tyrtyshnikov 2010], `TT-DMRG-cross`[Savostyanov & Oseledets 2011], `greedy2cross`[Savostyanov 2014]
- The best one to our best knowledge, `greedy2cross`, needs $\mathcal{O}(dnR^2)$ entries of $\mathcal{T}$.

Black box TT approximation

- New Goal: Approximate $\mathcal{T}$ in TT format using **as few function evaluations as possible** where

$$\mathcal{T}_{i_1,\ldots,i_d} = f(x^{(1)}_{i_1}, \ldots, x^{(d)}_{i_d}).$$

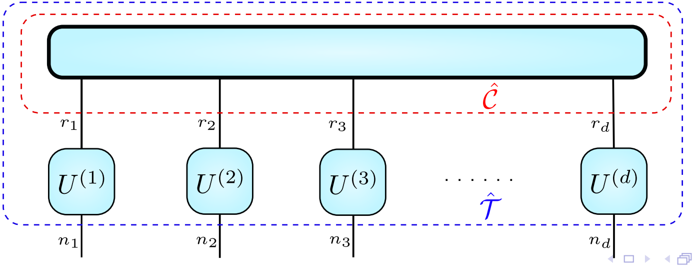


- Previous work: based on the Adaptive Cross Approximation (ACA) for matrices.
- Many variants of ACA for TT: `TT-cross`[Oseledets & Tyrtyshnikov 2010], `TT-DMRG-cross`[Savostyanov & Oseledets 2011], `greedy2cross`[Savostyanov 2014]
- The best one to our best knowledge, `greedy2cross`, needs $\mathcal{O}(dnR^2)$ entries of $\mathcal{T}$.
- $R \ll n$. Can we somehow reduce n further?

Black box TT approximation with an additional low-rank structure

- Novel idea: first approximate $\mathcal{T}$ in Tucker format (implicitly):

$$\mathcal{T} \approx \mathcal{C} \times_1 U^{(1)} \times_2 \cdots \times_d U^{(d)},$$

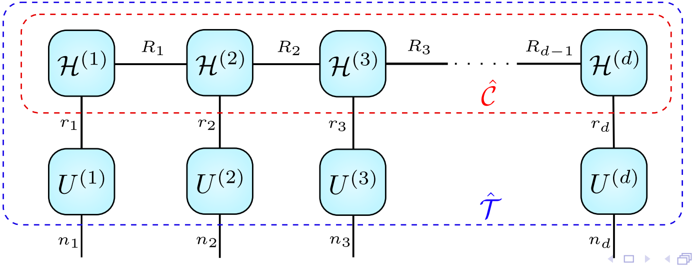


Black box TT approximation with an additional low-rank structure

- Novel idea: first approximate $\mathcal{T}$ in Tucker format (implicitly):

$$\mathcal{T} \approx \mathcal{C} \times_1 U^{(1)} \times_2 \cdots \times_d U^{(d)},$$

- then use `greedy2cross` to compute the TT approximation of $\mathcal{C}$.

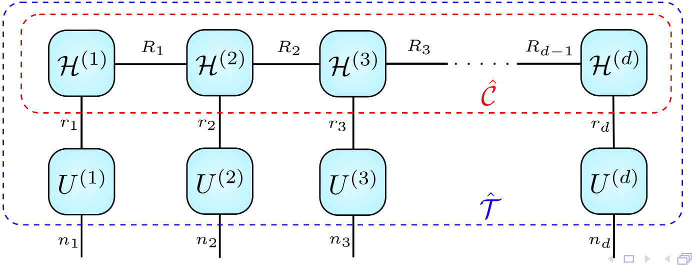


Black box TT approximation with an additional low-rank structure

- Novel idea: first approximate $\mathcal{T}$ in Tucker format (implicitly):

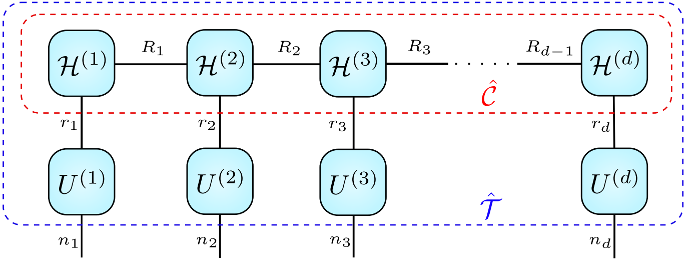
$$\mathcal{T} \approx \mathcal{C} \times_1 U^{(1)} \times_2 \cdots \times_d U^{(d)},$$

- then use `greedy2cross` to compute the TT approximation of $\mathcal{C}$.
- Applying `greedy2cross` to $\mathcal{C}$ only requires $\mathcal{O}(drR^2)$ entries, instead of $\mathcal{O}(dnR^2)$ for $\mathcal{T}$.



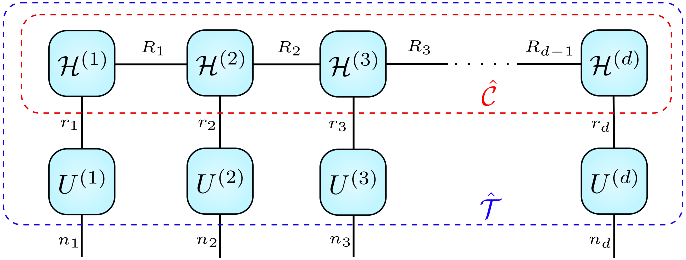
Black box TT approximation with an additional low-rank structure

- Novel idea: first approximate $\mathcal{T}$ in Tucker format (implicitly).
- then use `greedy2cross` to compute the TT approximation of $\mathcal{C}$.
- Applying `greedy2cross` to $\mathcal{C}$ only requires $\mathcal{O}(drR^2)$ entries,
  instead of $\mathcal{O}(dnR^2)$ for $\mathcal{T}$.
- Important: $\mathcal{C}$ must be constructed **implicitly**.



Black box TT approximation with an additional low-rank structure

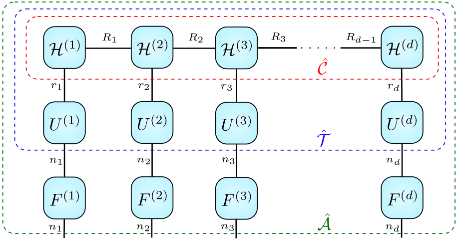
- Novel idea: first approximate $\mathcal{T}$ in Tucker format (implicitly).
- then use `greedy2cross` to compute the TT approximation of $\mathcal{C}$.
- Applying `greedy2cross` to $\mathcal{C}$ only requires $\mathcal{O}(drR^2)$ entries, instead of $\mathcal{O}(dnR^2)$ for $\mathcal{T}$.
- Important: $\mathcal{C}$ must be constructed **implicitly**.
- Our construction ensures $\mathcal{C}$ to be a **subtensor** of $\mathcal{T}$.



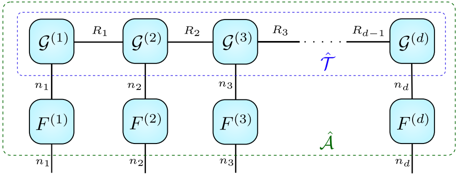
The extended functional tensor train (EFTT) format

$$f(x_1,\dots,x_d) \approx \hat{f}(x_1,\dots,x_d) = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \hat{\mathcal{A}}_{i_1,\dots,i_d} T_{i_1}(x_1)\cdots T_{i_d}(x_d)$$

$$\hat{f}(x_1,\dots,x_d) = \sum_{\alpha_1=1}^{R_1} \dots, \sum_{\alpha_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,\alpha_1}(x_1) g^{(2)}_{\alpha_1,\alpha_2}(x_2) \cdots g^{(d-1)}_{\alpha_{d-2},\alpha_{d-1}}(x_{d-1}) g^{(d)}_{\alpha_{d-1},1}(x_d)$$



(a) EFTT

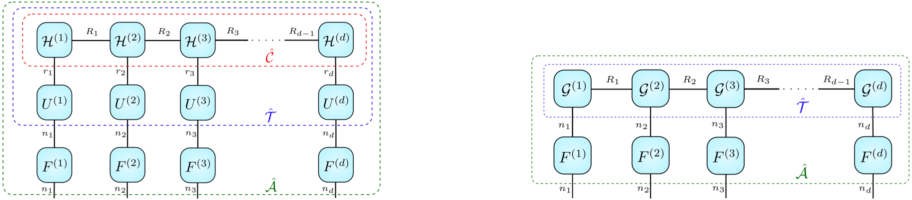


(b) FTT

The extended functional tensor train (EFTT) format

$$f(x_1,\dots,x_d) \approx \hat{f}(x_1,\dots,x_d) = \sum_{i_1=1}^{n_1}\cdots\sum_{i_d=1}^{n_d} \hat{\mathcal{A}}_{i_1,\dots,i_d} T_{i_1}(x_1)\cdots T_{i_d}(x_d)$$

$$\hat{f}(x_1,\dots,x_d) = \sum_{\alpha_1=1}^{R_1}\dots,\sum_{\alpha_{d-1}=1}^{R_{d-1}} g^{(1)}_{1,\alpha_1}(x_1) g^{(2)}_{\alpha_1,\alpha_2}(x_2)\cdots g^{(d-1)}_{\alpha_{d-2},\alpha_{d-1}}(x_{d-1}) g^{(d)}_{\alpha_{d-1},1}(x_d)$$



(a) EFTT (b) FTT

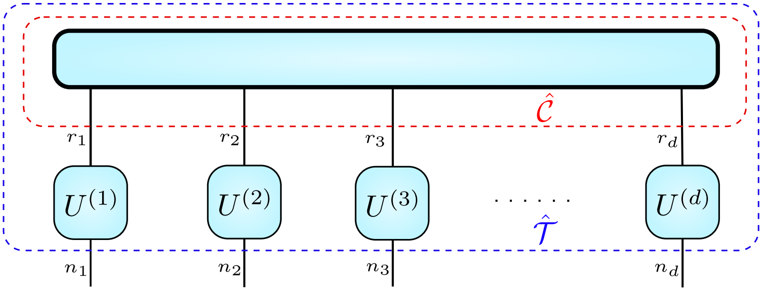
$$g^{(\ell)}_{\alpha_{\ell-1},\alpha_\ell}(x_\ell) = \sum_{k=1}^{n_\ell}\left(\mathcal{H}^{(\ell)} \times_2 U^{(\ell)} \times_2 F^{(\ell)}\right)_{\alpha_{\ell-1},k,\alpha_\ell} T_k(x_\ell),\ g^{(\ell)}_{\alpha_{\ell-1},\alpha_\ell}(x_\ell) = \sum_{k=1}^{n_\ell}\left(\mathcal{G}^{(\ell)} \times_2 F^{(\ell)}\right)_{\alpha_{\ell-1},k,\alpha_\ell} T_k(x_\ell)$$

Organization

1. Functional low-rank approximation
2. The extended functional tensor train (EFTT) format
3. EFTT approximation algorithm
4. Numerical experiments

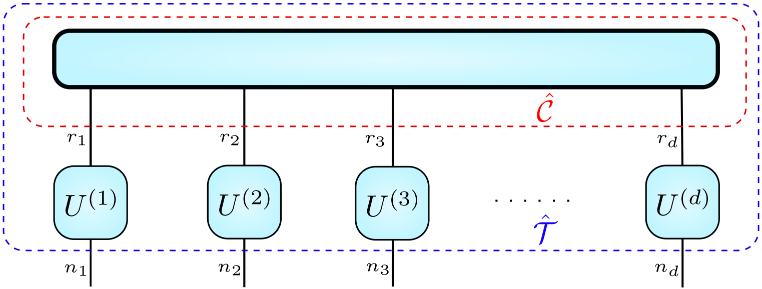
Step 1: Construct factor matrices $U^{(\ell)}$

- **Q:** What makes $U^{(\ell)}$ a good factor matrix?



Step 1: Construct factor matrices $U^{(\ell)}$

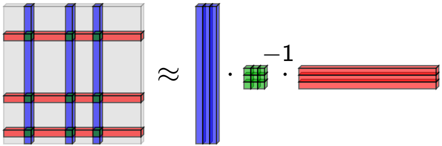
- **Q:** What makes $U^{(\ell)}$ a good factor matrix?
- **A:** (HOSVD[De Lathauwer, De Moor, Vandewalle 2000]) $\Longrightarrow$ $[U^{(\ell)}, \sim, \sim] = \texttt{svd}(\mathcal{T}^{\{\ell\}})$



Step 1: Construct factor matrices $U^{(\ell)}$

- **Q:** What makes $U^{(\ell)}$ a good factor matrix?
- **A:** (HOSVD[De Lathauwer, De Moor, Vandewalle 2000]) $\Longrightarrow$ $[U^{(\ell)}, \sim, \sim] = \mathtt{svd}(\mathcal{T}^{\{\ell\}})$
- ACA requires evaluating all entries of $\mathcal{T}^{\{\ell\}}$.

$$M \approx M(:,\mathcal{J})M(\mathcal{I},\mathcal{J})^{-1}M(\mathcal{I},:) \rightsquigarrow$$



$\mathcal{I} = \emptyset$, $\mathcal{J} = \emptyset$, $A = M$

while true **do**

    if $\max_{(i,j)} |A_{i,j}| \leq \varepsilon$ **then** return $\mathcal{I}, \mathcal{J}$ **end**

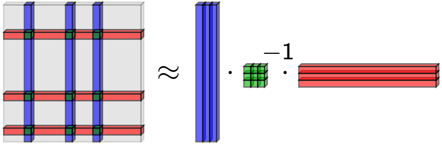
    $(i^*, j^*) = \arg\max_{(i,j)} |A_{i,j}|$, $\mathcal{I} = \mathcal{I} \cup \{i^*\}$, $\mathcal{J} = \mathcal{J} \cup \{j^*\}$

    $A = M - M(:,\mathcal{J})M(\mathcal{I},\mathcal{J})^{-1}M(\mathcal{I},:)$

Step 1: Construct factor matrices $U^{(\ell)}$

- **Q:** What makes $U^{(\ell)}$ a good factor matrix?
- **A:** (HOSVD[De Lathauwer, De Moor, Vandewalle 2000]) $\implies$ $[U^{(\ell)}, \sim, \sim] = \texttt{svd}(\mathcal{T}^{\{\ell\}})$
- ACA requires evaluating all entries of $\mathcal{T}^{\{\ell\}}$.
- We proposed a randomized pivoted ACA to avoid this.

$$M \approx M(:,\mathcal{J})M(\mathcal{I},\mathcal{J})^{-1}M(\mathcal{I},:) \rightsquigarrow$$

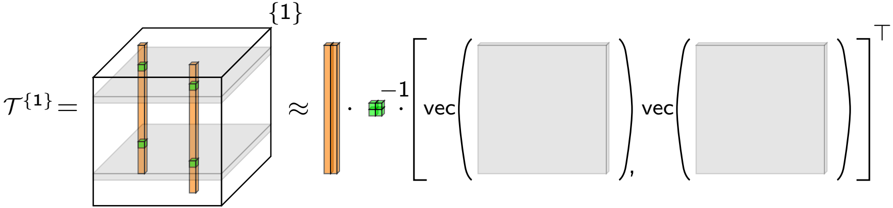


1: $\mathcal{I} = \emptyset$, $\mathcal{J} = \emptyset$, $A = M$
2: **while** true **do**
3: **if** $\max_{(i,j)\in S} |A_{i,j}| \leq \varepsilon$ **then** return $\mathcal{I}, \mathcal{J}$ **end**
4: Construct $S \subset \{1,\ldots,n\} \times \{1,\ldots,m\}$ by uniformly sampling s index pairs.
5: $(i^*, j^*) = \arg\max_{(i,j)\in S} |A_{i,j}|$, $\mathcal{I} = \mathcal{I} \cup \{i^*\}$, $\mathcal{J} = \mathcal{J} \cup \{j^*\}$
6: $A = M - M(:,\mathcal{J})M(\mathcal{I},\mathcal{J})^{-1}M(\mathcal{I},:)$

Step 1: Construct factor matrices $U^{(\ell)}$

- **Q:** What makes $U^{(\ell)}$ a good factor matrix?
- **A:** (HOSVD[De Lathauwer, De Moor, Vandewalle 2000]) $\implies [U^{(\ell)}, \sim, \sim] = \texttt{svd}(\mathcal{T}^{\{\ell\}})$
- ACA requires evaluating all entries of $\mathcal{T}^{\{\ell\}}$.
- We proposed a randomized pivoted ACA to avoid this.
- Applying RPACA to the mode-ℓ matricization $\mathcal{T}^{\{\ell\}} \in \mathbb{R}^{n_\ell \times n_1 \cdots n_{\ell-1} n_{\ell+1} \cdots n_d}$:

$$\mathcal{T}^{\{\ell\}} \approx \underbrace{\mathcal{T}^{\{\ell\}}(:, J_\ell)}_{U^{(\ell)}} (\mathcal{T}^{\{\ell\}}(I_\ell, J_\ell))^{-1} \mathcal{T}^{\{\ell\}}(I_\ell, :)$$

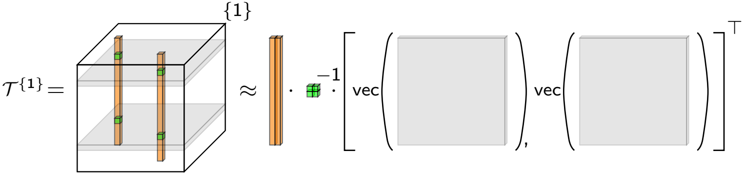


Step 1: Construct factor matrices $U^{(\ell)}$

Applying RPACA to the mode-ℓ matricization $\mathcal{T}^{\{\ell\}} \in \mathbb{R}^{n_\ell \times n_1 \cdots n_{\ell-1} n_{\ell+1} \cdots n_d}$:

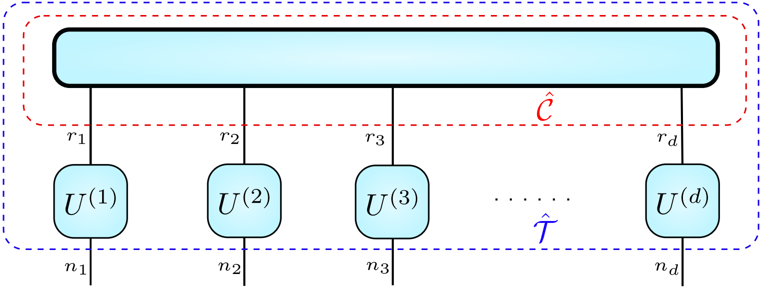
$$\mathcal{T}^{\{\ell\}} \approx \underbrace{\mathcal{T}^{\{\ell\}}(:, J_\ell)}_{U^{(\ell)}} (\mathcal{T}^{\{\ell\}}(I_\ell, J_\ell))^{-1} \mathcal{T}^{\{\ell\}}(I_\ell, :)$$

Advantages: 1. Only requires evaluating $\mathcal{O}(dr^3 + dsr^2 + dnr)$ entries of $\mathcal{T}$.
2. Adaptivity in Tucker rank r_ℓ's naturally (by choosing ACA tol ε).
3. Adaptivity in polynomial degree n_ℓ's by leveraging `Chebfun` heuristics.



Step 2: Construct the core tensor $\mathcal{C}$ implicitly

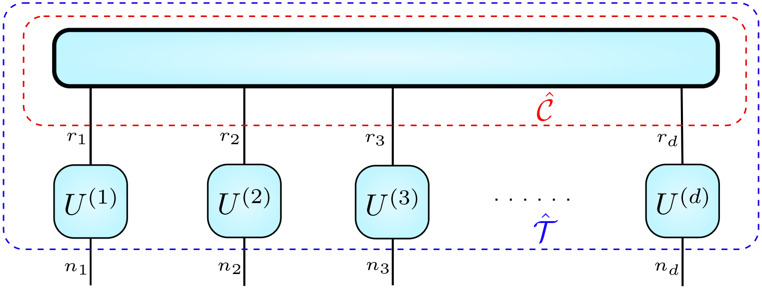
- **Q:** Given $U^{(\ell)}$ and $\mathcal{T}$, what's best $\mathcal{C}$?



Step 2: Construct the core tensor $\mathcal{C}$ implicitly

- **Q:** Given $U^{(\ell)}$ and $\mathcal{T}$, what's best $\mathcal{C}$?
- **A:** (HOSVD[De Lathauwer, De Moor, Vandewalle 2000]) $\Longrightarrow$ **Orthogonal projection** of $\mathcal{T}$ onto span$(U^{(\ell)})$, which requires full evaluation of $\mathcal{T}$, for $[Q^{(\ell)}, \sim] = \texttt{qr}(U^{(\ell)}, \texttt{"econ"})$:

$$\mathcal{T} \approx \underbrace{(\mathcal{T} \times_1 Q^{(1)^\top} \times_2 \cdots \times_d Q^{(d)^\top})}_{\mathcal{C}} \times_1 Q^{(1)} \times_2 \cdots \times_d Q^{(d)}.$$

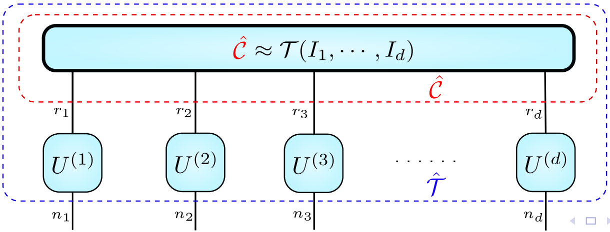


Step 2: Construct the core tensor $\mathcal{C}$ implicitly

- **Q:** Given $U^{(\ell)}$ and $\mathcal{T}$, what's best $\mathcal{C}$?
- **Orth. proj.**: $\mathcal{T} \approx \underbrace{(\mathcal{T} \times_1 Q^{(1)^\top} \times_2 \cdots \times_d Q^{(d)^\top})}_{\mathcal{C}} \times_1 Q^{(1)} \times_2 \cdots \times_d Q^{(d)}$.
- We propose to use an **oblique projection** $Q^{(\ell)}(\Phi_{I_\ell}^\top Q^{(\ell)})^{-1}\Phi_{I_\ell}^\top$:

$$\mathcal{T} \approx \underbrace{(\mathcal{T} \times_1 \Phi_{I_1}^\top \times_2 \cdots \times_d \Phi_{I_d}^\top)}_{\mathcal{C}=\mathcal{T}(I_1,\cdots,I_d)} \times_1 \underbrace{Q^{(1)}(\Phi_{I_1}^\top Q^{(1)})^{-1}}_{\text{updated } U^{(1)}} \times_2 \cdots \times_d \underbrace{Q^{(d)}(\Phi_{I_d}^\top Q^{(d)})^{-1}}_{\text{updated } U^{(d)}}$$

- Φ_{I_ℓ} is a sampling matrix s.t. $\Phi_{I_\ell}^\top Q^{(\ell)} = Q^{(\ell)}(I_\ell, :)$.

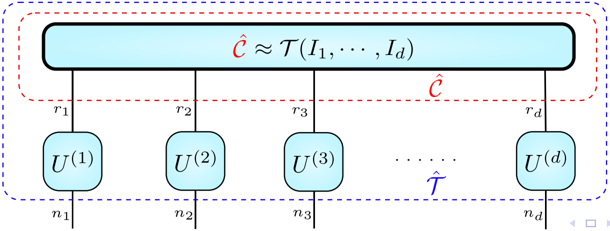


Step 2: Construct the core tensor $\mathcal{C}$ implicitly

- **Oblique projection** $Q^{(\ell)}(\Phi_{I_\ell}^\top Q^{(\ell)})^{-1}\Phi_{I_\ell}^\top$:

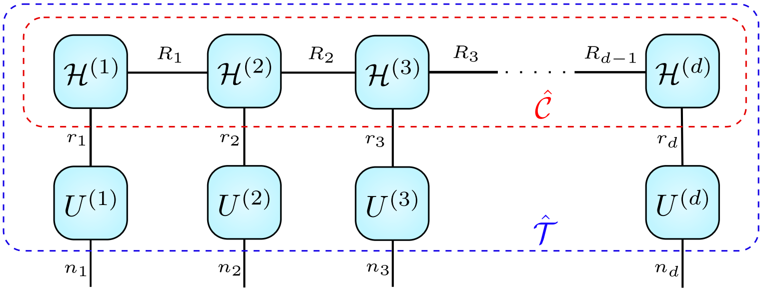
$$\mathcal{T} \approx \underbrace{(\mathcal{T} \times_1 \Phi_{I_1}^\top \times_2 \cdots \times_d \Phi_{I_d}^\top)}_{\mathcal{C}=\mathcal{T}(I_1,\cdots,I_d)} \times_1 \underbrace{Q^{(1)}(\Phi_{I_1}^\top Q^{(1)})^{-1}}_{\text{updated } U^{(1)}} \times_2 \cdots \times_d \underbrace{Q^{(d)}(\Phi_{I_d}^\top Q^{(d)})^{-1}}_{\text{updated } U^{(d)}}$$

- Φ_{I_ℓ} is a sampling matrix s.t. $\Phi_{I_\ell}^\top Q^{(\ell)} = Q^{(\ell)}(I_\ell,:)$.
- Error introduced by the oblique projection $\sim \|Q^{(\ell)}(I_\ell,:)^{-1}\|_2$ can be minimized by selecting I_ℓ carefully, e.g., using model order reduction methods like `DEIM`.
- **No evaluation** of $\mathcal{T}$ in Step 2.



Step 3: Compute the TT approximation of $\mathcal{C}$

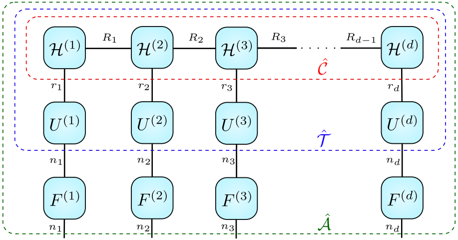
- We apply `greedy2cross` to $\mathcal{C} = \mathcal{T}(I_1, \cdots, I_d)$ to obtain the TT format of $\mathcal{C}$.
- Requires $\mathcal{O}(drR^2)$ entries of $\mathcal{T}$.



EFTT approximation algorithm summary

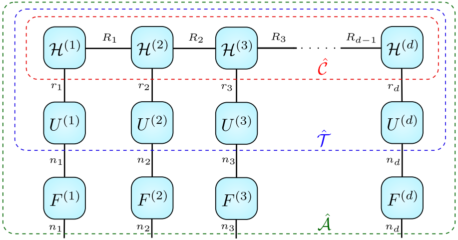
1. Define a procedure to evaluate the entries of $\mathcal{T}$.
2. Construct $U^{(\ell)}$ using RPACA.
3. Update $U^{(\ell)}$ and define $\mathcal{C}$ implicitly by oblique projection.
4. Compute the TT approximation of $\mathcal{C}$ using `greedy2cross`.
5. Define procedures to evaluate the univariate functions:

$$g^{(\ell)}_{\alpha_{\ell-1},\alpha_\ell}(x_\ell) = \sum_{j=1}^{n_\ell} \left(\mathcal{H}^{(\ell)} \times_2 U^{(\ell)} \times_2 F^{(\ell)} \right)_{\alpha_{\ell-1},j,\alpha_\ell} T_j(x_\ell).$$



EFTT approximation algorithm summary

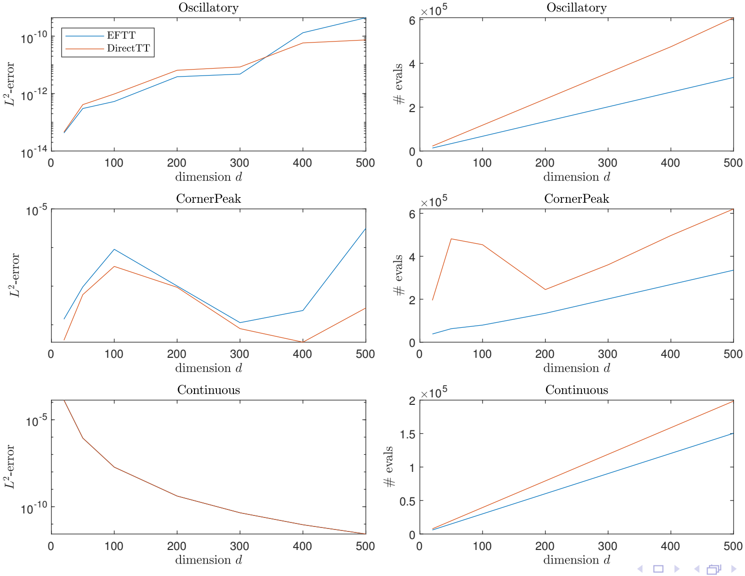
- EFTT approximation algorithm:
  - Step 1: Construct $U^{(\ell)}$ using RPACA.
  - Step 2: Update $U^{(\ell)}$ and define $\mathcal{C}$ implicitly by oblique projection.
  - Step 3: Compute the TT approximation of $\mathcal{C}$ using `greedy2cross`.
- Needs $\mathcal{O}(\underbrace{dr^3 + dsr^2}_{\text{RPACA}} + \underbrace{dnr}_{\text{evaluating } \hat{U}^{(\ell)}} + \underbrace{drR^2}_{\text{greedy2cross applied to } \mathcal{C}})$ entries of $\mathcal{T}$.
- Comparing to applying `greedy2cross` directly to $\mathcal{T}$: $\mathcal{O}(dnR^2)$.



Organization

1. Functional low-rank approximation
2. The extended functional tensor train (EFTT) format
3. EFTT approximation algorithm
4. Numerical experiments

Compare with directly applying greedy2cross to $\mathcal{T}$ for Genz functions:



Compare with directly applying greedy2cross to $\mathcal{T}$ for benchmark functions from science & engineering

Function (d)	Algorithm	Error	# func. evals	# dofs	$\max_\ell n_\ell$	$\max_\ell R_\ell$	$\max_\ell r_\ell$
Piston (7)	**EFTT**	3.32e-09	203484	74228	100	24	11
	DirectTT	2.93e-09	992566	412603	100	18	
Borehole (8)	**EFTT**	3.95e-02	14186	3243	100	2	4
	DirectTT	3.95e-02	10042	2318	100	2	
OTL Circuit (6)	**EFTT**	3.71e-11	16065	3280	100	5	5
	DirectTT	8.49e-12	27764	8300	100	4	
Robot Arm (8)	**EFTT**	7.00e-02	500591	101847	100	33	33
	DirectTT	6.52e-02	734573	383466	100	34	
Wing Weight (10)	**EFTT**	3.73e-14	6692	2072	100	2	2
	DirectTT	8.29e-14	10440	3600	100	2	
Friedman (5)	**EFTT**	4.41e-10	12317	2377	100	4	4
	DirectTT	8.84e-12	14676	3142	100	3	
G & L (6)	**EFTT**	2.52e-05	3278	1034	100	2	2
	DirectTT	2.52e-05	6651	1800	100	2	
G & P 8D (8)	**EFTT**	3.08e-11	39724	8138	100	7	7
	DirectTT	2.64e-11	74942	30140	100	5	
D & P Exp (3)	**EFTT**	1.56e-14	1990	616	100	2	2
	DirectTT	1.55e-14	2087	800	100	2	

Compare with continuous analog of TT-cross (c3py) [Gorodetsky, Karaman, Marzouk 2019] for benchmark functions from science & engineering

Function (d)	Algorithm	Error	# func. evals	# dofs	$\max_\ell n_\ell$	$\max_\ell R_\ell$	$\max_\ell r_\ell$
Piston (7)	**EFTT**	3.71e-09	174188	69019	33	23	11
	c3py	3.85e-05	251760	66080	35	24	
Borehole (8)	**EFTT**	3.95e-02	6552	1116	32	2	4
	c3py	2.08e-03	14346	577	70	2	
OTL Circuit (6)	**EFTT**	7.93e-11	6670	1083	27	5	5
	c3py	4.07e-08	15674	1782	28	5	
Robot Arm (8)	**EFTT**	8.12e-02	499954	54760	94	12	27
	c3py	3.85e-01	2018017	228439	105	20	
Wing Weight (10)	**EFTT**	2.83e-14	2867	560	24	2	2
	c3py	2.15e-13	12224	554	19	2	
Friedman (5)	**EFTT**	2.16e-11	5238	404	19	3	4
	c3py	8.08e-05	12142	710	15	4	
G & L (6)	**EFTT**	4.95e-06	1547	356	29	2	2
	c3py	3.51e-02	13928	374	105	2	
G & P 8D (8)	**EFTT**	4.77e-11	19527	3902	24	6	7
	c3py	9.54e-10	27336	5136	21	7	
D & P Exp (3)	**EFTT**	1.13e-14	2404	646	105	2	2
	c3py	4.78e-10	12162	336	49	2	

Summary

- Paper available at https://arxiv.org/abs/2211.11338
- Code available at https://github.com/bonans/EFTT