Fast Tensor Product Schwarz Smoothers

Julius Witte Daniel Arndt Guido Kanschat

Heidelberg University

Interdisciplinary Center for Scientific Computing

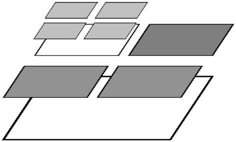
Sixth `deal.II` Users and Developers Workshop

July 23-27, 2018

Overview

SPPEXA - ExaDG

Massive parallelism will facilitate extreme computing up to exascale
$\rightarrow$ adapt finite element solvers



- Matrix-free implementations based on tensor product polynomials[1]
  - memory efficient
  - high computational intensity
- Higher order DG methods

- (Adaptive) geometric multigrid preconditioners with Schwarz smoothers
  - powerful tool, robustness
  - very expansive for higher order
  - memory inefficient (contradictory to matrix-free)

[1] M. Kronbichler and K. Kormann. "A generic interface for parallel cell-based finite element operator application". In: *Computers & Fluids* 63 (2012), pp. 135–147

Motivation

- Access to main memory is very slow compared to arithmetic operations on cache memory, e.g. matrix-vector products
- Standard approach:

$$a(u, v) \xrightarrow{\text{store}} \text{sparse matrix } A \in \mathbb{R}^{N \times N} \xrightarrow{\text{apply}} x = Au$$

- Matrix-free approach:

$$a(u, v) \xrightarrow{\text{routine}} \text{linear operator } F : \mathbb{R}^N \to \mathbb{R}^N \xrightarrow{\text{apply}} x = F(u)$$

- SSOR, ILU and AMG are not applicable
- Geometric multigrid is solely based on matrix-vector products (transfer, smoother)

Tensor Product Elements

Tensor Product Shape Functions

Let $\hat{K} = [0,1]^d$ be reference cell and $V^{(t)} \subset \{\phi : [0,1] \to \mathbb{R}\}$ a n_t-dimensional Hilbert space. We define $V(\hat{K}) := \bigotimes_{t=1}^{d} V^{(t)}$ by

$$\hat{\phi}^{(1)}(\hat{x}_1) \otimes \ldots \otimes \hat{\phi}^{(d)}(\hat{x}_d) := \hat{\phi}^{(1)}(\hat{x}_1)\hat{\phi}^{(2)}(\hat{x}_2) \cdots \hat{\phi}^{(d)}(\hat{x}_d).$$

Let $\{\hat{\varphi}_i = \hat{\phi}^{(1)}_{i_1} \cdots \hat{\phi}^{(d)}_{i_d}\}$ denote a set of basis functions with lexicographical multi-index $i = i_1 + \sum_{t=2}^{d} i_t \prod_{\tau=1}^{t-1} n_t$.

- Assume isotropy, i.e. $n = n_1 = \ldots = n_d$
- d-fold Cartesian product $\hat{K} \;\to\;$ tensor product quadrature
- Number of quadrature points is a constant multiple of n

Sum-factorization

- $F_K : \hat{K} \to K$ denotes cell transformation
- v_h denotes (global) FE basis function

Local FE Expansion

Evaluation in each quadrature point requires $\mathcal{O}(n^{2d})$ arithmetic operations,

$$v_h|_K(\mathbf{x}_q) = \sum_{j=1}^{n^d} v_j^K \hat{\varphi}_j(\hat{\mathbf{x}}_q), \quad \mathbf{x}_q = F_K(\hat{\mathbf{x}}_q).$$

Sum-factorization

Exploiting tensor products facilitates reduction to $\mathcal{O}(dn^{d+1})$ operations,

$$v_h|_K(\mathbf{x}_q) = \sum_{j_d=1}^{n} \hat{\phi}_{j_d}(\hat{x}_{q_d}) \cdots \sum_{j_2=1}^{n} \hat{\phi}_{j_2}(\hat{x}_{q_2}) \sum_{j_1=1}^{n} v_{j_1,\dots,j_d}^K \hat{\phi}_{j_1}(\hat{x}_{q_1}).$$

Generic Setting

- Multi-index $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ and $\boldsymbol{\beta}$ handling derivatives
- $\partial^\alpha \hat{\phi}$ is αth derivative of the univariate function $\hat{\phi}$
- $\kappa(x_1, \ldots, x_d) = \kappa^{(1)}(x_1) \cdots \kappa^{(d)}(x_d)$ is a separable coefficient
- $\Xi(x_1, \ldots, x_d)$ is a non-separable coefficient

Generic Tensor Product Setting

Assume F_K allows an element-based stiffness matrix A^K in the form

$$A^K_{ij} = \sum_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma} \sum_{q_2} \partial^{\alpha_2} \hat{\phi}_{j_2}(\hat{x}_{q_2}) \partial^{\beta_2} \hat{\phi}_{i_2}(\hat{x}_{q_2}) \kappa^{(2)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_2})$$
$$\sum_{q_1} \partial^{\alpha_1} \hat{\phi}_{j_1}(\hat{x}_{q_1}) \partial^{\beta_1} \hat{\phi}_{i_1}(\hat{x}_{q_1}) \kappa^{(1)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_1}) \Xi_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_1}, \hat{x}_{q_2}).$$

Example: Laplace

Numerical integration of the Laplacian in weak form,

$$A_{ij}^K = \int_K \boldsymbol{\nabla}\varphi_j^K \cdot \boldsymbol{\nabla}\varphi_i^K \, \mathrm{d}\mathbf{x}$$

$$= \sum_{q_1, q_2} \left(\hat{\mathrm{J}}_K^{-T} \; \hat{\boldsymbol{\nabla}}\hat{\varphi}_j \cdot \hat{\mathrm{J}}_K^{-T} \; \hat{\boldsymbol{\nabla}}\hat{\varphi}_i \left| \det \hat{\mathrm{J}}_K \right| \right) (\hat{x}_{q_1}, \hat{x}_{q_2}) \, w_{q_1} w_{q_2}.$$

Generic setting obtained by means of

- $\boldsymbol{\alpha}_1 = \boldsymbol{\beta}_1 = (1,0)$ and $\boldsymbol{\alpha}_2 = \boldsymbol{\beta}_2 = (0,1)$,
- $\kappa_{\boldsymbol{\alpha}_k, \boldsymbol{\beta}_l} \equiv \kappa$ encapsulates quadrature weights w_{q_1}, w_{q_2},
- $\Xi_{\boldsymbol{\alpha}_k, \boldsymbol{\beta}_l} = (\hat{\mathrm{J}}_K^{-1} \, \hat{\mathrm{J}}_K^{-T} \left| \det \hat{\mathrm{J}}_K \right|)_{kl}, \; k, l \in \{1, 2\}$.

Fast Finite Element Operator Application

- $u^K \in \mathbb{R}^{n^d}$ denotes local degrees of freedom
- A^K has generic tensor product form

$$\sum_j A^K_{ij} u^K_j = \sum_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma} \sum_{q_2} \partial^{\beta_2} \hat{\phi}_{i_2}(\hat{x}_{q_2}) \kappa^{(2)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_2}) \sum_{q_1} \partial^{\beta_1} \hat{\phi}_{i_1}(\hat{x}_{q_1}) \kappa^{(1)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_1}) \Xi_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_1}, \hat{x}_{q_2}) \tilde{u}_{\boldsymbol{\alpha},q_1,q_2}, \tag{1}$$

where

$$\tilde{u}_{\boldsymbol{\alpha},q_1,q_2} = \sum_{j_2} \partial^{\alpha_2} \hat{\phi}_{j_2}(\hat{x}_{q_2}) \sum_{j_1} \partial^{\alpha_1} \hat{\phi}_{j_1}(\hat{x}_{q_1}) u^K_{j_1,j_2}. \tag{2}$$

- eq. (2) by `FEEvaluation::gather_evaluate()`
- eq. (1) by `*::submit_value()/_gradient()/...` + `*::integrate()`

Kronecker Decomposition

Kronecker Decomposition

We are interested in the best approximation of *Kronecker rank r*,

$$A^K \approx \sum_{i=1}^{r} A_i^{(d)} \otimes \cdots \otimes A_i^{(1)},$$

where $\otimes$ denotes the Kronecker product. Here $A_i^{(t)}$ is a $n \times n$ matrix.

If A^K is given in the generic tensor product setting,

$$A^K = \sum_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma} \sum_{q_d} \cdots \sum_{q_2} A^{(d)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma,q_d} \otimes \cdots \otimes A^{(2)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma,q_2} \otimes A^{(1)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma,q_d,\dots,q_2},$$

with

$$\left(A^{(1)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma,q_d,\dots,q_2}\right)_{ij} = \sum_{q_1} \partial^{\alpha_1}\hat{\phi}_j(\hat{x}_{q_1})\partial^{\beta_1}\hat{\phi}_i(\hat{x}_{q_1})\kappa_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_1})\Xi_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_1},\dots,\hat{x}_{q_d}),$$

$$\left(A^{(t)}_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma,q_t}\right)_{ij} = \partial^{\alpha_t}\hat{\phi}_j(\hat{x}_{q_t})\partial^{\beta_t}\hat{\phi}_i(\hat{x}_{q_t})\kappa_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}(\hat{x}_{q_t}), \quad t = 2,\dots,d.$$

Separable Kronecker Decomposition

- $r \leq Cn^d$, with generic constant C dependant on $\boldsymbol{\alpha}, \boldsymbol{\beta}, \gamma$
- $\Xi_{\boldsymbol{\alpha},\boldsymbol{\beta},\gamma}$ prevents dimension-independant bound C

$\Rightarrow$ Separability is the key

separable Kronecker Decomposition[2]

Fast inversion algorithms rely on a separable Kronecker decomposition,

$$A = M^{(d)} \otimes \cdots \otimes M^{(2)} \otimes A^{(1)} + \ldots + A^{(d)} \otimes M^{(d-1)} \otimes \cdots \otimes M^{(1)},$$

We refer to $A_i^{(t)}$ and $M_i^{(t)}$, here $n \times n$ matrices, as the *action* and *basic state*, respectively.

[2] R.E. Lynch, J.R. Rice, and D.H. Thomas. "Direct solution of partial difference equations by tensor product methods". In: *Numerische Mathematik* 6.1 (1964), pp. 185–199

Fast Diagonalization Method (FDM)

For convenience, assume A is symmetric.

Generalized Eigenvalue Problems

Find eigenvectors, denoted as the columns of the orthogonal matrix $Z^{(t)}$, that satisfy

$$(Z^{(t)})^T A^{(t)} Z^{(t)} = \Lambda^{(t)}, \qquad (Z^{(t)})^T M^{(t)} Z^{(t)} = I^{(t)},$$

where $I^{(t)}$ is the identity matrix of appropriate size.

Fast Diagonalization Method

Basis transformation of A into the eigenvector spaces,

$$(Z^{(2)} \otimes Z^{(1)})^T A \, (Z^{(2)} \otimes Z^{(1)}) = I^{(2)} \otimes \Lambda^{(1)} + \Lambda^{(2)} \otimes I^{(1)},$$

leads fast inversion in diagonal form,

$$A^{-1} = Z^{(2)} \otimes Z^{(1)} \left[I^{(2)} \otimes \Lambda^{(1)} + \Lambda^{(2)} \otimes I^{(1)} \right]^{-1} (Z^{(2)} \otimes Z^{(1)})^T.$$

FDM - Reduced Complexities

Matrix-vector products regarding A^{-1} in Kronecker product form facilitates sum-factorization twice.

- *Inverting.* solve 1D generalized eigenvalue problems.
- *Storage.* d matrices of eigenvectors and d vectors of eigenvalues.
- *Multiplication.* Kronecker decomposition of basis transformation facilitates sum-factorization.

	standard	*tensor product*
inverting	$\mathcal{O}(n^{3d})$	$\mathcal{O}(dn^3)$
storage	$\mathcal{O}(n^{2d})$	$\mathcal{O}(dn^2)$
vmult	$\mathcal{O}(n^{2d})$	$\mathcal{O}(dn^{d+1})$

Table: Complexity comparison to standard inverting

- `TensorProductMatrixSymmetricSum` performs FDM (if A is symmetric)

Abstract Schwarz Theory

Let V be Hilbert space, $f \in V'$. Find $u \in V$ such that

$$a(u, v) = f(v) \quad \forall v \in V,$$

where $a(\, . \, , \, . \,)$ is symmetric, positive definite (s.p.d.).

Schwarz Operators

Assume V admits decomposition $\sum_{s \in \mathcal{S}} R_s^T V_s$. Then, Schwarz operators $P_s \equiv R_s^T \tilde{P}_s$ are defined by projection operators,

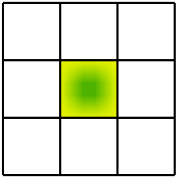
$$\tilde{a}_s(\tilde{P}_s u, v_s) = a(u, R_s^T v_s) \quad \forall v_s \in V_s,$$

where $\tilde{a}_s(.,.)$ are local bilinear forms (s.p.d.).

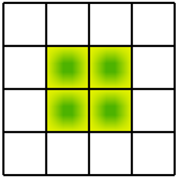
An *exact local bilinear form* is given by

$$\tilde{a}_s(u_s, v_s) \equiv a(R_s^T u_s, R_s^T v_s), \quad u_s, v_s \in V_s.$$

Schwarz Smoothers



(a) cell patch



(b) (regular) vertex patch

The *additive* preconditioned system is defined as

$$P_{ad} = \sum_{s \in \mathcal{S}} P_s = \Big(\sum_{s \in \mathcal{S}} R_s^T \tilde{A}_s^{-1} R_s\Big) A,$$

and the colored *multiplicative* preconditioned system as

$$P_{mu} = I - \Big(I - \sum_{s \in \mathcal{S}_C} P_s\Big) \cdots \Big(I - \sum_{s \in \mathcal{S}_1} P_s\Big),$$

where the index set $\mathcal{S}_c$ gathers all subdomains of color $c = 1, \dots, C$.

Separable Subproblems

- Geometrical decomposition $\Omega = \bigcup_{s \in \mathcal{S}} \Omega_s$

Idea: Bilinear forms $\tilde{a}_s(\, . \, , \, . \,)$ that lead to separable Kronecker decompositions $\tilde{A}_s$ approximating the exact local stiffness matrix A^{Ω_s}

Local Stability

There exists $\hat{\gamma} > 0$ such that

$$a(R_s^T u_s, R_s^T u_s) \leq \hat{\gamma} \, \tilde{a}_s(u_s, u_s), \; u_s \in range(\tilde{P}_s) \subset V_s, \; s \in \mathcal{S}.$$

- *Local stability* ensures coercivity of subproblems
- Additional damping $\hat{\omega} \sim \hat{\gamma}^{-1}$ is necessary

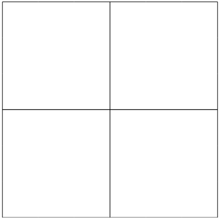
Numerical Experiment

Poisson Problem (non-homogeneous)

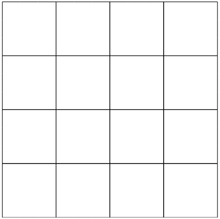
$$-\Delta u = f \text{ in } \Omega, \quad u = g \text{ on } \partial\Omega,$$

- *Discretization*:
  - symmetric interior penalty method (SIPG)
  - discontinuous tensor product polynomials $\mathbb{Q}_k^{DG}$
- *Solver*: residual reduction of 10^{-8}
  - PCG for symmetric preconditioned systems
  - and GMRES for non-symmetric
- *Multigrid Preconditioner*:
  - standard V-cycle
  - $m = 1$ pre- and post-smoothing step

Test Settings - Cartesian



(a) Coarse grid.



(b) Uniformly refined.

- *Exact Subproblem*:
  - mapping preserves tensor product structure
- *Graph Coloring*[3]:
  - schedules tasks equally
  - not minimizing amount of colors

[3] B. Turcksin, M. Kronbichler, and W. Bangerth. "*WorkStream* – a design pattern for multicore-enabled finite element computations". In: *ACM Transactions on Mathematical Software* 43.1 (2016), pp. 2/1–2/29

Cell Patch Smoothers

Levels	Convergence steps			
2D	$k=3$	$k=4$	$k=7$	$k=10$
5	15	15	19	21
6	15	15	19	21
7	15	15	19	21
8	15	15	19	21
9	15	15	19	21
10	15	15	19	21
3D	$k=3$	$k=4$	$k=7$	$k=10$
2	16	16	21	24
3	17	17	22	25
4	18	17	22	25
5	18	18	23	25
6	18	18	23	25

(a) Additive. $\omega = 0.7$.

Levels	Convergence steps			
2D	$k=3$	$k=4$	$k=7$	$k=10$
5	8	8	11	11
6	8	8	10	11
7	8	8	10	11
8	8	8	10	11
9	8	8	10	11
10	8	8	10	11
3D	$k=3$	$k=4$	$k=7$	$k=10$
2	8	9	12	13
3	9	9	12	13
4	9	9	12	13
5	9	9	12	13
6	9	9	12	13

(b) Multiplicative[4]. $\omega = 1$.

- Cell patch smoothers are robust w.r.t. mesh size h.

[4] Red-Black coloring.

Multiplicative Vertex Patch Smoothers

Levels	Convergence steps			
2D	$k=3$	$k=4$	$k=7$	$k=10$
6	2.9	3.0	2.6	2.5
7	2.9	2.9	2.6	2.5
8	2.9	2.9	2.6	2.5
9	2.9	2.9	2.6	2.5
10	2.9	2.9	2.6	2.4
3D	$k=3$	$k=4$	$k=7$	$k=10$
3	2.6	2.7	2.4	2.3
4	2.8	2.8	2.5	2.3
5	2.8	2.8	2.5	2.4
6	2.8	2.8	2.5	2.4

(a) Graph coloring.

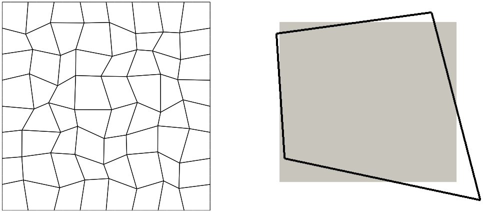
Levels	Convergence steps			
2D	$k=3$	$k=4$	$k=7$	$k=10$
6	2.5	2.5	2.1	1.9
7	2.5	2.5	2.1	1.9
8	2.5	2.5	2.1	1.8
9	2.5	2.4	2.1	1.8
10	2.5	2.4	2.0	1.8
3D	$k=3$	$k=4$	$k=7$	$k=10$
3	2.4	2.5	2.1	1.8
4	2.4	2.5	2.1	1.9
5	2.4	2.5	2.1	1.8
6	2.4	2.5	2.1	1.8

(b) Manual coloring.[5]

- Multiplicative vertex patch smoother is robust w.r.t. to mesh size h and polynomial degree k.
- Coloring is not negligible concerning iteration steps.

[5] Minimal amount of 8 colors in 2D and 16 colors in 3D.

Test Settings - Unstructured



(a) Coarse grid.[6] (b) Surrogate Cell.

- *Inexact Subproblem*.
  - solve on surrogate patches (Cartesian)
  - subproblems have to be additionally damped $\hat{\omega}$
- $\mathbb{Q}_4^{DG}$ polynomials.

[6] Illustrative. Coarse grid contains 1024 cells in 2D and 512 in 3D.

Additive Smoothers (inexact)

Levels	Convergence steps						
2D	*exact*	$\hat{\omega} = 0.80$	$\hat{\omega} = 0.75$	$\hat{\omega} = 0.70$	$\hat{\omega} = 0.65$	$\hat{\omega} = 0.60$	$m = 2$
2	28	35	34	33	34	36	23
3	28	43	39	37	38	40	24
4	27	45	40	37	38	42	24
5	27	47	41	38	39	44	23
6	27	n.c.	43	39	40	45	23
3D	*exact*	$\hat{\omega} = 0.85$	$\hat{\omega} = 0.80$	$\hat{\omega} = 0.75$	$\hat{\omega} = 0.70$	$\hat{\omega} = 0.65$	$m = 2$
2	31	39	37	37	37	39	26
3	32	43	40	39	41	44	26
4	32	43	41	40	43	47	26

Table: Additive cell patch smoother. $\omega = 0.7\,\hat{\omega}$.

- Inexact additive smoothers are robust w.r.t. mesh size h for sufficiently many smoothing steps m.

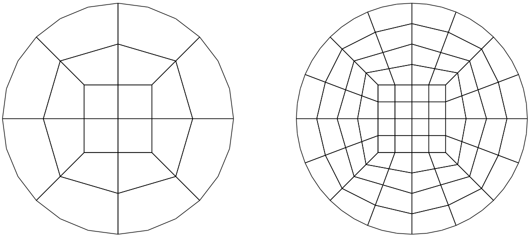
Multiplicative Smoothers (inexact)

Levels	Convergence steps					
2D	$\hat{\omega} = 0.95$	$\hat{\omega} = 0.90$	$\hat{\omega} = 0.85$	$\hat{\omega} = 0.80$	$\hat{\omega} = 0.75$	$\hat{\omega} = 0.70$
2	27	17	17	18	18	19
3	16	17	18	18	19	20
4	20	17	17	18	19	20
5	36	17	17	17	18	19
6	41	17	17	17	18	18
3D	$\hat{\omega} = 0.85$	$\hat{\omega} = 0.80$	$\hat{\omega} = 0.75$	$\hat{\omega} = 0.70$	$\hat{\omega} = 0.65$	$\hat{\omega} = 0.60$
2	21	21	22	23	24	25
3	20	21	22	23	24	25
4	34	25	25	23	23	24

(a) Multiplicative cell patch smoother. $\omega = \hat{\omega}$.

- Inexact multiplicative smoothers are robust w.r.t. mesh size h.
- *No convergence for vertex patches possible.*

Test Settings - Curved Boundary



(a) Coarse grid. (b) Uniformly refined.

- *Inexact Subproblem*:
  - solve on surrogate patches (Cartesian)
  - subproblems have to be additionally damped $\hat{\omega}$
- $\mathbb{Q}_4^{DG}$ polynomials and isoparametric mapping.

Cell Patch Smoothers (inexact)

Levels	Convergence steps						
2D	*exact*	$\hat{\omega} = 0.90$	$\hat{\omega} = 0.85$	$\hat{\omega} = 0.80$	$\hat{\omega} = 0.75$	$\hat{\omega} = 0.70$	$m = 2$
6	30	39	37	36	38	42	23
7	30	42	38	37	40	43	23
8	32	44	39	37	40	43	23
9	31	46	40	37	40	43	23
10	31	48	41	38	40	43	23

(a) Additive cell patch smoother. $\omega = 0.7\,\hat{\omega}$.

Levels	Convergence steps					
2D	$\hat{\omega} = 0.90$	$\hat{\omega} = 0.85$	$\hat{\omega} = 0.80$	$\hat{\omega} = 0.75$	$\hat{\omega} = 0.70$	$\hat{\omega} = 0.65$
6	n.c.	18	18	19	20	21
7	n.c.	19	18	19	20	21
8	n.c.	18	17	18	19	20
9	n.c.	20	17	18	19	20
10	n.c.	20	17	17	18	19

(b) Multiplicative cell patch smoother. $\omega = \hat{\omega}$.

- Inexact cell patch smoothers are robust w.r.t. mesh size h for sufficiently many smoothing steps m.

Nearest Separable Coefficient

- How to treat non-separable (physical) coefficients Ξ, e.g. diffusion coefficient?

Nearest Separable Coefficient

Determine the best approximation

$$\Xi \circ F_K(\hat{\boldsymbol{x}}) \approx \hat{\xi}^{(1)}(\hat{x}_1) \cdots \hat{\xi}^{(d)}(\hat{x}_d).$$

Idea: Compute the best rank-1 tensor $\tilde{\mathcal{X}} = \tilde{\boldsymbol{x}}^{(1)} \otimes \cdots \otimes \tilde{\boldsymbol{x}}^{(d)}$ minimizing $\left\| \mathcal{X} - \tilde{\mathcal{X}} \right\|$ with $\mathcal{X}_{q_1,\dots,q_d} = \Xi \circ F_K(\hat{\boldsymbol{x}}_q)$.

- Alternating least squares (ALS) is slow
- No guaranteed convergence to global minimum
- $\tilde{\mathcal{X}}$ heavily depends on initial guess

FDM - non-symmetric case

Multiplying the separable Kronecker decomposition A by $(M^{(d)} \otimes \cdots \otimes M^{(1)})^{-1}$ results in

$$\bar{A} = I^{(d)} \otimes \cdots \otimes I^{(2)} \otimes (M^{(1)})^{-1} A^{(1)} + \ldots + (M^{(d)})^{-1} A^{(d)} \otimes I^{(d-1)} \otimes \cdots \otimes I^{(1)}$$

SVD

There exist orthogonal matrices $U^{(t)}$, $V^{(t)}$ and a diagonal matrix $\Sigma^{(t)}$ such that

$$(M^{(t)})^{-1} A^{(t)} = U^{(t)} \Sigma^{(t)} (V^{(t)})^T.$$

Fast Diagonalization Method

The SVDs result in a fast inversion of $\bar{A}$,

$$\bar{A}^{-1} = V^{(2)} \otimes V^{(1)} \left[I^{(2)} \otimes \Sigma^{(1)} + \Sigma^{(2)} \otimes I^{(1)} \right]^{-1} (U^{(2)} \otimes U^{(1)})^T.$$

- Matrix-vector products regarding $A^{-1} = \bar{A}^{-1}(M^{(d)} \otimes \cdots \otimes M^{(1)})^{-1}$ facilitate three successive sum-factorizations