Tensor Approximations for Elliptic PDEs with Jumping Coefficients

Sergey Dolgov[1]

[1]Moscow Institute of Physics and Technology www.mipt.ru
Institute of Numerical Mathematics, Moscow, Russia www.inm.ras.ru

Tensor Methods in Multi-Dimensional Boundary-Value and Spectral problems
MPI MiS Leipzig, May 3-5, 2010

Introduction

Main problem

Diffusion equation

$$\begin{cases} -\nabla(a(\mathbf{x})\nabla u(\mathbf{x})) = f(\mathbf{x}) \quad in \quad \Omega \subset \mathbb{R}^d \\ \alpha u(\mathbf{x})|_{\partial\Omega} + \beta \frac{\partial u}{\partial \mathbf{n}}|_{\partial\Omega} = g(\partial\Omega), \end{cases} \tag{1}$$

where $a > 0$, $\alpha^2 + \beta^2 \neq 0$.

Examples

- Flow models: heat conductivity, liquid, gas flows.
- Electrostatics.
- Financial math.

Solution methods - GFEM

Galerkin method ("weak", generalized formulation) ($g \equiv 0$, $\beta = 0$)

- Find u: $(a\nabla u, \nabla\phi)_{L_2(\Omega)} = (f, \phi)_{L_2(\Omega)}$.
- u and ϕ are assumed to belong to some function class in H^1.

Discrete form - FEM

- suppose $u = \sum_{\mathbf{i}} u_{\mathbf{i}}\phi_{\mathbf{i}}(\mathbf{x})$, $\phi_{\mathbf{i}} \in H^1$.
- find matrix $\Gamma = \left[(a\nabla\phi_{\mathbf{i}}, \nabla\phi_{\mathbf{j}})\right]$,
- right-part vector $F = (F_1, ..., F_N)^T$, $F_{\mathbf{i}} = (f, \phi_{\mathbf{i}})$.
- Solve $\Gamma\mathbf{u} = F$, where $\mathbf{u} = (u_1, ..., u_N)^T$.

GFEM

Computational difficulties

- Curse of dimensionality:
  d dimensions, n grid points in each variable. Then $N = n^d$.
- Ill conditioned matrices:
  $\mathtt{cond}(\Gamma) \sim 10^8 - 10^9$.

GFEM

Computational difficulties

- Curse of dimensionality:
  d dimensions, n grid points in each variable. Then $N = n^d$.
- Ill conditioned matrices:
  $\mathrm{cond}(\Gamma) \sim 10^8 - 10^9$.

Approaches

- Use of data compression tensor technics.
- Use of preconditioners, special solvers.

Continuous formulation

Simple problem

$$\begin{cases} -\nabla(a\nabla u) = f \quad in\ \Omega = [0,1]^d \\ u\big|_{\partial\Omega} = 0. \end{cases}$$

Continuous formulation

Simple problem

$$\begin{cases} -\nabla(a\nabla u) = f \quad in\ \Omega = [0,1]^d \\ u|_{\partial\Omega} = 0. \end{cases}$$

Auxiliary Poisson equation

$$\begin{cases} -\Delta v = f \quad in\ \Omega = [0,1]^d \\ v|_{\partial\Omega} = 0. \end{cases}$$

Continuous formulation

Simple problem

$$\begin{cases} -\nabla(a\nabla u) = f \quad in\ \Omega = [0,1]^d \\ u|_{\partial\Omega} = 0. \end{cases}$$

Auxiliary Poisson equation

$$\begin{cases} -\Delta v = f \quad in\ \Omega = [0,1]^d \\ v|_{\partial\Omega} = 0. \end{cases}$$

"Motivating" equation

Suppose v is known. Then

$$-\nabla(a\nabla u) = -\Delta v.$$

This formulation brings good stuff

Discrete formulation

Galerkin formulation

- Consider $u_h = \sum_{\mathbf{i}} u_{\mathbf{i}} \phi_{\mathbf{i}}(\mathbf{x})$, $v_h = \sum_{\mathbf{i}} v_{\mathbf{i}} \phi_{\mathbf{i}}(\mathbf{x})$, $\{\phi_{\mathbf{i}}\} \in H^1(\Omega)$.
- We are to solve $(a\nabla u_h, \nabla \phi_{\mathbf{j}}) = (\nabla v_h, \nabla \phi_{\mathbf{j}})$.

Where are tensors?

- Suppose $\phi_{\mathbf{i}}(\mathbf{x}) = \varphi_{i_1}(x_1) \cdots \varphi_{i_d}(x_d)$.
- Then

$$\Delta_h = [(\nabla \phi_{\mathbf{i}}, \nabla \phi_{\mathbf{j}})] = G \otimes H \otimes \cdots \otimes H + \cdots + H \otimes \cdots \otimes H \otimes G,$$

$G = [(\nabla \varphi_i, \nabla \varphi_j)]$ - 1D stiffness, $H = [(\varphi_i, \varphi_j)]$ - mass matrix.

Discrete formulation

Why tensors?

- By imposing separability properties on a and f, we can also get separability for v and u.
- Hence we get $O(nd)$ rather than $O(n^d)$ complexity.

Requirements

Suppose the following conditions:

- We have $(a\nabla u_h, \nabla \phi_{\mathbf{j}}) = (\nabla v_h, \nabla \phi_{\mathbf{j}})$.
- $\phi_{\mathbf{i}}(\mathbf{x}) = \varphi_{i_1}(x_1) \cdots \varphi_{i_d}(x_d)$, $i_q = 1, ..., n$, $q = 1..d$, $\mathrm{supp}(\varphi_i) \in [x_{i-1}, x_{i+1}]$.

Requirements

Suppose the following conditions:

- We have $(a\nabla u_h, \nabla \phi_{\mathbf{j}}) = (\nabla v_h, \nabla \phi_{\mathbf{j}})$.
- $\phi_{\mathbf{i}}(\mathbf{x}) = \varphi_{i_1}(x_1) \cdots \varphi_{i_d}(x_d)$, $i_q = 1, ..., n$, $q = 1..d$, $\mathrm{supp}(\varphi_i) \in [x_{i-1}, x_{i+1}]$.
- $v_{\mathbf{i}} \approx v_{r,\mathbf{i}} = \sum\limits_{k=1}^{r_v} v^{(1)}_{k,i_1} \cdots v^{(d)}_{k,i_d}$, $||v - v_r|| \leq \varepsilon_v$

Requirements

Suppose the following conditions:

- We have $(a\nabla u_h, \nabla\phi_{\mathbf{j}}) = (\nabla v_h, \nabla\phi_{\mathbf{j}})$.
- $\phi_{\mathbf{i}}(\mathbf{x}) = \varphi_{i_1}(x_1)\cdots\varphi_{i_d}(x_d),\ i_q = 1, ..., n,\ q = 1..d$, $\mathrm{supp}(\varphi_i) \in [x_{i-1}, x_{i+1}]$.
- $v_{\mathbf{i}} \approx v_{r,\mathbf{i}} = \sum\limits_{k=1}^{r_v} v^{(1)}_{k,i_1} \cdots v^{(d)}_{k,i_d},\ ||v - v_r|| \leq \varepsilon_v$
- $\dfrac{1}{a_{\mathbf{i}}} \approx \dfrac{1}{a_{r,\mathbf{i}}} = \sum\limits_{l=1}^{r_{1/a}} \dfrac{1}{a^{(1)}_{l,i_d}} \cdots \dfrac{1}{a^{(d)}_{l,i_1}},\ ||\dfrac{1}{a} - \dfrac{1}{a_r}|| \leq \varepsilon_a$

Requirements

Suppose the following conditions:

- We have $(a\nabla u_h, \nabla \phi_{\mathbf{j}}) = (\nabla v_h, \nabla \phi_{\mathbf{j}})$.
- $\phi_{\mathbf{i}}(\mathbf{x}) = \varphi_{i_1}(x_1)\cdots\varphi_{i_d}(x_d),\ i_q = 1, ..., n,\ q = 1..d,$ $\mathrm{supp}(\varphi_i) \in [x_{i-1}, x_{i+1}]$.
- $v_{\mathbf{i}} \approx v_{r,\mathbf{i}} = \sum\limits_{k=1}^{r_v} v^{(1)}_{k,i_1}\cdots v^{(d)}_{k,i_d},\ ||v - v_r|| \leq \varepsilon_v$
- $\dfrac{1}{a_{\mathbf{i}}} \approx \dfrac{1}{a_{r,\mathbf{i}}} = \sum\limits_{l=1}^{r_{1/a}} \dfrac{1}{a^{(1)}_{l,i_d}}\cdots\dfrac{1}{a^{(d)}_{l,i_1}},\ ||\dfrac{1}{a} - \dfrac{1}{a_r}|| \leq \varepsilon_a$
- $a^{(q)}_{l,i} > 0,\ q = 1, ..., d,\ l = 1, ..., r_{1/a},\ i = 1, ..., n$

Result

Then we get:

- $u_{\mathbf{i}} \approx u_{r,\mathbf{i}} = \sum\limits_{k=1}^{r_u} u^{(1)}_{k,i_1} \cdots u^{(d)}_{k,i_d}$, $||u - u_r|| \leq \varepsilon_u$.

Result

Then we get:

- $u_{\mathbf{i}} \approx u_{r,\mathbf{i}} = \sum_{k=1}^{r_u} u^{(1)}_{k,i_1} \cdots u^{(d)}_{k,i_d}$, $||u - u_r|| \leq \varepsilon_u$.
- $\varepsilon_u = O(\varepsilon_v) + O(\varepsilon_a) + O(h^\alpha)$, $h = 1/(n+1)$, $\alpha = 1, 2$.
- $r_u = r_{1/a} r_v$.

Result

Then we get:

- $u_{\mathbf{i}} \approx u_{r,\mathbf{i}} = \sum_{k=1}^{r_u} u_{k,i_1}^{(1)} \cdots u_{k,i_d}^{(d)}, \; ||u - u_r|| \leq \varepsilon_u.$
- $\varepsilon_u = O(\varepsilon_v) + O(\varepsilon_a) + O(h^\alpha), \; h = 1/(n+1), \; \alpha = 1, 2.$
- $r_u = r_{1/a} r_v.$
- Factors $u^{(q)}$ can be computed independently from the systems with tridiagonal matrices:

$$a_{i-1} \; u_{i-1} + \frac{a_{i-1} + a_i}{2} F_1(v_r) \; u_i + a_i \; u_{i+1} = \frac{1}{a_i} F_0(v_r)$$

Black-box preconditioner

Left preconditioning

- We are to solve $Ax = f \leftarrow$ hard.
- Apply a non singular matrix B: $BAx = Bf$.
- Solve $(BA)x = (Bf)$.

Black-box MatVec approach

- Iterative solvers exploit just Ax operation.
- Obtaining BA and $(BA)x$ can be difficult.
- One may apply black-box procedure for $Ax \rightarrow y$ and $By \rightarrow z$ on each iteration (faster).

Black-box preconditioner

Application to diffusion problem

- Suppose $Au = f$ - diffusion problem with separability properties.
- We have an algorithm which gives $\tilde{u} \approx u = A^{-1}f$:
  - Compute $v = \Delta^{-1}f$ (using FFT, quadratures - fast).
  - Apply sweep-based algorithm: $\tilde{u} = S[1/a](v)$.
- Now we have $By \to z$ operation: $z = S[1/a](\Delta^{-1}y)$.
- FFT, quadrature solver, sweep solver - fast methods, complexity $O(n)$, $O(n \ \log n)$.

As A is sparse, we have
complexity of one iteration of $O(n)$, or $O(n \ \log n)$.

Numerical results

- We test complexity $S[1/a](v)$ on v in canonical format with rank r_v, $\mathrm{rank}(1/a) = r_{1/a}$, grid size n.

Table: CPU time versus n, $r_v = 59$, $r_{1/a} = 1$, $d = 2$.

n	CPU Time, sec
32	0.069945
64	0.139432
128	0.275668
256	0.515547
512	1.005040
1024	1.987609

Numerical results

- We test complexity $S[1/a](v)$ on v in canonical format with rank r_v, $\mathrm{rank}(1/a) = r_{1/a}$, grid size n.

Table: CPU time versus $r_u = r_{1/a} r_v$, $d = 2$, $n = 256$

r_u	CPU Time, sec
8	0.081331
16	0.147973
32	0.283700
64	0.562467

Numerical results

- We test complexity $S[1/a](v)$ on v in canonical format with rank r_v, $\mathrm{rank}(1/a) = r_{1/a}$, grid size n.

Table: CPU time versus d, $r_u = 16$, $n = 256$

d	CPU Time, sec
2	0.147973
4	0.522492
8	3.120825
16	23.232763

The complexity is of $O(d^3)$ (due to $F_1(v_r)$, $F_0(v_r)$).
Application on full vector $N = n^d$ is of $O(N)$.

Gradient equation

- Recall $(a\nabla u, \nabla\phi) = (\nabla v, \nabla\phi) \Rightarrow (a\nabla u - \nabla v, \nabla\phi) = 0$, $\phi \in H^1$.
- Require $\nabla u = \frac{1}{a}\nabla v$ in the least squares sence: $||\nabla u - \frac{1}{a}\nabla v||^2 \to \min.$
- Then $\Delta u - \nabla\frac{1}{a}\nabla v = 0.$
- If $v = \Delta^{-1}f$, then $u = \Delta^{-1}(\nabla\frac{1}{a}\nabla)\Delta^{-1}f.$

Gradient equation

- Suppose $\Gamma[a]u = f$ is a diffusion problem with coefficient a, $\Gamma[a]u = \nabla(a\nabla u)$.
- Then $\Gamma[1/a] = \nabla(\frac{1}{a}\nabla)$.
- We have the following preconditioner: $\Delta^{-1}\Gamma[1/a]\Delta^{-1}$

$\Delta^{-1}\Gamma[1/a]\Delta^{-1} \approx \Gamma[a]^{-1}$?

- Hypothesis: $\Delta^{-1}\Gamma[1/a]\Delta^{-1}\ \Gamma[a] = I + R$.
- In 1D: $\mathrm{rank}\, R = 1$ (in functional view: $\dim R = 1$) (proved).
- In higher dimension: R has low-rank approximation (in functional view: R is a compact operator) (hypothesis).

LS functional vs Black-box discrete

LS

- (+) Simple explicit operator form.
- (+) Symmetric matrix.
- ($\pm$) Convergence depends on jumping of the coefficient.
- (-) Requires 2 Laplace inversions (4(3) FFTs).
- (-) $||\Gamma[a]^{-1} - \Delta^{-1}\Gamma[1/a]\Delta^{-1}|| = O(1)$.
- (-) In tensor form, has the rank $\mathrm{rank}(1/a)\,\mathrm{rank}(\Delta^{-1})^2$.

LS functional vs Black-box discrete

Black-box

- (+) Requires 1 Laplace inversion + $d \cdot \mathrm{rank}$ sweeps.
- (+) Intrinsic tensor structure with ranks $\mathrm{rank}(1/a)\,\mathrm{rank}(\Delta^{-1})$.
- (+) $||\Gamma[a]^{-1} - S(\Delta^{-1})|| = O(h^{\alpha})$.
- ($\pm$) Convergence depends on rank of $1/a$.
- (-) Non-symmetric matrix.
- (-) Requires positive tensor approximation for $1/a$.
- (-) Complicated formulation.

Common properties

- We are solving diffusion problem via GMRES with one of mentioned preconditioners.
- Initial guess is taken as zero in all cases.
- For integration in Galerkin matrix assembly, the rectangle quadrature formula is used.
- No restarts are made in GMRES.
- For both preconditioners, the low-rank positive diffusion is used.

Dependence on n

Smooth coefficient: $1/a = (x+1)\cdot \mathbf{e}^y + 2\cos(x+y)$

Table: Number of iterations to $||Au - f||/||f|| < 10^{-8}$, and CPU time of one iteration.

n	Black-box	time, s	LS	time,s
32	5	0.0024	3	0.002261
64	5	0.0079	3	0.005482
128	5	0.0293	3	0.02683
256	5	0.1065	3	0.093763
512	5	0.5008	3	0.513589

Dependence on n

Smooth coefficient: $1/a = (x+1)\cdot \mathbf{e}^y + 2\cos(x+y)$

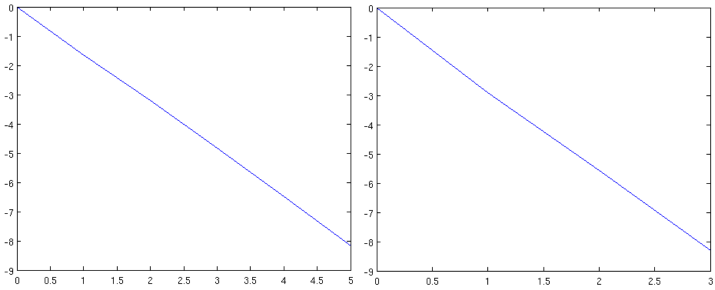

Figure: Convergence history for Black-box (left) and LS (right):

$\log \frac{||Au-f||}{||f||}$ vs number of iteration.

Dependence on n

Jumping coefficient: $1/a$ of rank 3, values from 2 to 1400.

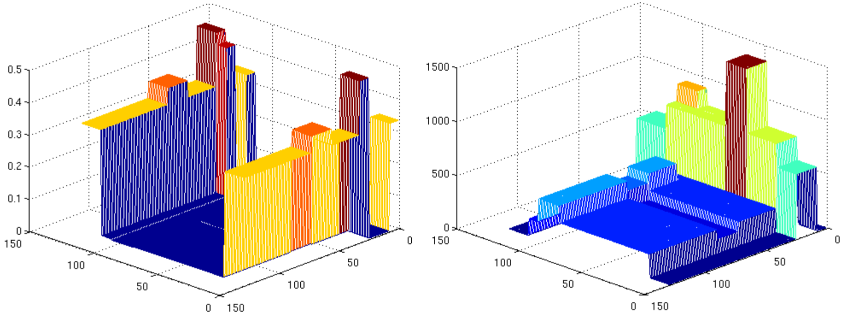

Figure: Diffusion coefficient a (left) and $1/a$ (right)

Dependence on n

Jumping coefficient: $1/a$ of rank 3, values from 2 to 1400.

Table: Number of iterations to $||Au - f||/||f|| < 10^{-8}$, and CPU time of one iteration.

n	Black-box	time, s	LS	time, s
32	17	0.0022	45	0.002394
64	18	0.0063	51	0.005692
128	18	0.0259	53	0.023622
256	18	0.0999	54	0.098719
512	18	0.5275	55	0.512126

Dependence on n

Jumping coefficient: $1/a$ of rank 3, values from 2 to 1400.

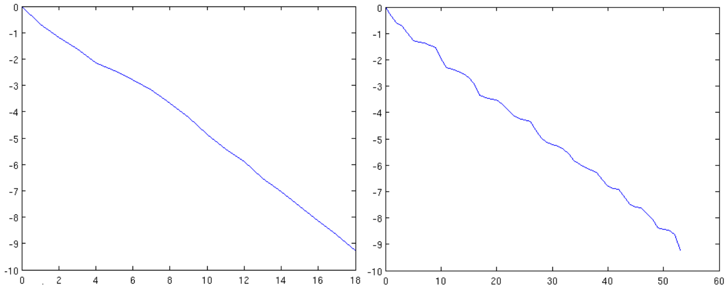

Figure: Convergence history for Black-box (left) and LS (right):

$$\log \frac{||Au - f||}{||f||} \text{ vs number of iteration.}$$

Dependence on jumps and ranks

Rank-2 checkerboard: $1/a = \mathtt{chk}(x) \cdot 1 + 1 \cdot \mathtt{chk}(y)$, where

$$\mathtt{chk}(x) = \begin{cases} 1, [x \cdot 16] \ \textit{is odd}, \\ \alpha, [x \cdot 16] \ \textit{is even}. \end{cases}$$

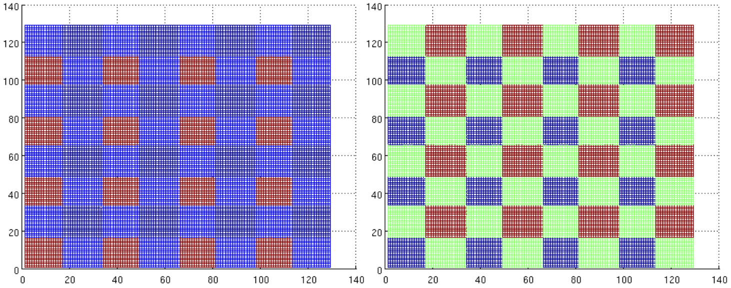

Figure: Diffusion coefficient a (left) and $1/a$ (right)

Dependence on jumps and ranks

Rank-2 checkerboard: dependency on α, $n = 128$.

Table: Number of iterations to $||Au - f||/||f|| < 10^{-8}$.

α	Black-box	LS
0.01	14	14
0.1	10	8
10	11	9
100	19	17
1000	23	24

CPU time of one iteration: Black-box - 0.0207 s, LS - 0.024795 s.

Dependence on jumps and ranks

Rank-2 checkerboard: $\alpha = 1000.$

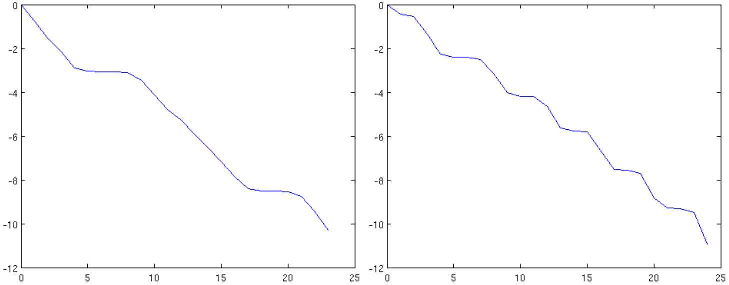

Figure: Convergence history for Black-box (left) and LS (right):

$$\log \frac{||Au-f||}{||f||}$$ vs number of iteration.

Dependence on jumps and ranks

3 test functions:

- $1/a_1 = \mathtt{if1}(x)\cdot\mathtt{if2}(y)\ (\mathrm{rank}=1)$,
- $1/a_2 = \mathtt{if1}(x)\cdot\mathtt{if2}(y) + (x+1)\cdot\mathbf{e}^y\ (\mathrm{rank}=2)$,
- $1/a_3 = \mathtt{if1}(x)\cdot\mathtt{if2}(y) + (x+1)\cdot\mathbf{e}^y + \mathtt{chk}(x)\cdot 1 + 1\cdot\mathtt{chk}(y)$ $(\mathrm{rank}=4)$,

where

$$\mathtt{if1}(x) = \begin{cases} 0.1, & x\in[0,0.25], \\ 1, & x\in(0.25,1]; \end{cases} \qquad \mathtt{if2}(x) = \begin{cases} 1, & x\in[0,0.75), \\ 7, & x\in[0.75,1], \end{cases}$$

$\mathtt{chk}(x)$ parameter $\alpha = 10$.

Dependence on jumps and ranks

3 test functions:

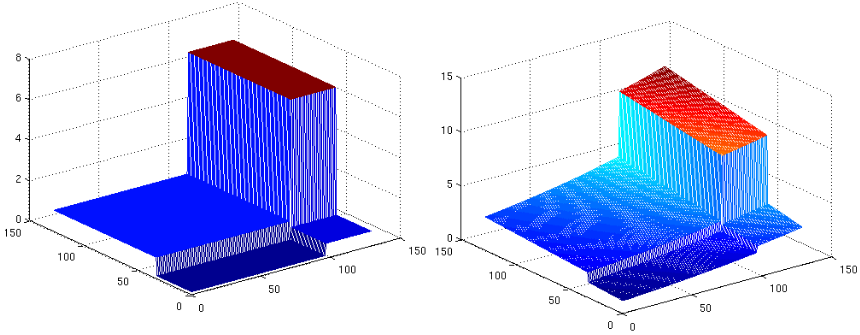

Figure: Reciprocal coefficients $1/a_1$ (left) and $1/a_2$ (right)

Dependence on jumps and ranks

3 test functions: dependency on rank, $n = 128$.

Table: Number of iterations to $||Au - f||/||f|| < 10^{-8}$, and CPU time of one iteration.

coefficient	Black-box	time, s	LS	time, s
a_1 $(\mathrm{rank} = 1,\ \frac{\max}{\min} = 70)$	6	0.017	12	0.023088
a_2 $(\mathrm{rank} = 2,\ \frac{\max}{\min} = 11.3)$	8	0.022	6	0.024139
a_3 $(\mathrm{rank} = 4,\ \frac{\max}{\min} = 10.4)$	13	0.029	8	0.022821

Dependence on jumps and ranks

3 test functions: $a = a_1$.

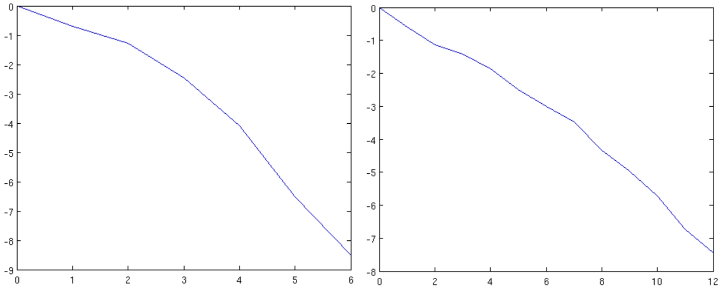

Figure: Convergence history for Black-box (left) and LS (right):

$$\log \frac{||Au - f||}{||f||}$$ vs number of iteration.

3D Dirichlet sample problems

$$f = 1.\ n = 64.$$

- $1/a_1 = \mathtt{if1}(x) \cdot \mathtt{if2}(y) \cdot \mathtt{if3}(z)$,

$$\mathtt{if3}(x) = \begin{cases} 10, & x \in [0.125, 0.25], \\ 1, & otherwise; \end{cases}$$

- $1/a_2 = \mathtt{if1}(x) \cdot \mathtt{if2}(y) \cdot \mathtt{if3}(z) + (x+1) \cdot \mathbf{e}^y \cdot \dfrac{1}{1+10z}$;
- $1/a_3 = \mathtt{chk}(x) + \mathtt{chk}(y) + \mathtt{chk}(z)$, $\alpha = 10^{-3}$.

3D Dirichlet sample problems

$$f = 1.\ n = 64.$$

Table: Number of iterations to $||Au - f||/||f|| < 10^{-8}$, and CPU time of one iteration.

coefficient	Black-box	time, s	LS	time, s
a_1	12	0.5478	24	0.758098
a_2	12	0.7107	17	0.746054
a_3	21	0.8062	23	0.744929

3D Neumann problems

$$f = \cos(\pi x)\cos(\pi y)\cos(\pi z).\ \ n = 64.$$

- $1/a_1 = \mathtt{if1}(x) \cdot \mathtt{if2}(y) \cdot \mathtt{if3}(z),$

$$\mathtt{if3}(x) = \begin{cases} 10, & x \in [0.125, 0.25], \\ 1, & otherwise; \end{cases}$$

- $1/a_2 = \mathtt{if1}(x) \cdot \mathtt{if2}(y) \cdot \mathtt{if3}(z) + (x+1) \cdot \mathbf{e}^y \cdot \dfrac{1}{1+10z};$
- $1/a_3 = \mathtt{chk}(x) + \mathtt{chk}(y) + \mathtt{chk}(z),\ \alpha = 10^{-3}.$

3D Neumann problems

$$f = \cos(\pi x)\cos(\pi y)\cos(\pi z). \ n = 64.$$

Table: Number of iterations to $||Au - f||/||f|| < 10^{-8}$, and CPU time of one iteration.

coefficient	Black-box	time, s	LS	time, s
a_1	13	0.6168	27	0.841716
a_2	25	0.7268	30	0.830226
a_3	15	0.8690	18	0.835752

Sample problem

Sample problem from the Society of Petroleum Engineers benchmark. $n = 512$, $\max(a) = 1000$, $\min(a) = 1$.

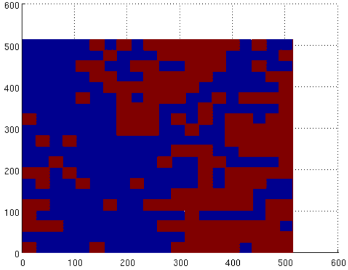

Figure: Diffusion coefficient a (left) and solution u (right)

LS preconditioner: 29 iterations, time of one iteration 0.526171 s.

- Proposed and compared two preconditioners for diffusion problem.
- Preconditioners can be applied in canonical tensor format, or full format, with complexity $O(n)$ and $O(N)$, correspondingly.
- For each coefficient can be chosen an optimal algorithm.

Future plans:

- Application to other PDEs (convection/reaction, etc; arbitrary domains).
- Further improvements of algorithms.

Thank you for your attention.