



MAX PLANCK INSTITUTE  
FOR DYNAMICS OF COMPLEX  
TECHNICAL SYSTEMS  
MAGDEBURG



COMPUTATIONAL METHODS IN  
SYSTEMS AND CONTROL THEORY

# Low-rank tensor methods for PDE-constrained optimization under uncertainty

Peter Benner

Joint work with

Sergey Dolgov (U Bath)

Martin Stoll (TU Chemnitz)

Akwum Onwunta (U Maryland)

Università di Pisa

April 4, 2019

Supported by:



OTTO VON GUERICKE  
UNIVERSITÄT  
MAGDEBURG



CDS  
Forschungszentrum  
Dynamische Systeme



## The Max Planck Society

- is dedicated to fundamental research;
- operates 84 institutes — 79 in Germany, 2 in Italy (Rome, Florence), 1 each in The Netherlands, Luxembourg, US;
- has ~ 23,000 employees;
- had 18 Nobel Laureates since 1948.

*"The first MPI in engineering. . ."*

- founded 1998
- 4 departments (directors)
- 10 research groups
- budget ~ 15 Mio. EUR
- ~ 230 employees
- ~ 160 scientific staff,
- doing research in
  - biotechnology
  - chemical engineering
  - process engineering
  - energy conversion
  - applied math
  - HPC



1. Introduction
2. Unsteady Heat Equation
3. Unsteady Navier-Stokes Equations
4. Conclusions



## 1. Introduction

Big data in flow control

Low-rank solvers for high-dimensional problems

PDE-constrained optimization under uncertainty

## 2. Unsteady Heat Equation

## 3. Unsteady Navier-Stokes Equations

## 4. Conclusions



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**
  - we use a grid with  $10^6$  nodes, for each node we store 4 real numbers: the velocity in  $x$ -,  $y$ -, and  $z$ -direction, plus the pressure;



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**
  - we use a grid with  $10^6$  nodes, for each node we store 4 real numbers: the velocity in x-, y-, and z-direction, plus the pressure;
  - we need 1,000 time steps to reach steady-state;



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**
  - we use a grid with  $10^6$  nodes, for each node we store 4 real numbers: the velocity in x-, y-, and z-direction, plus the pressure;
  - we need 1,000 time steps to reach steady-state;
  - we have 10 different design parameters, and want to test 10 values for each of them  $\rightsquigarrow 10^{10}$  configurations, i.e.,  $10^{10}$  transient simulations.



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**
  - we use a grid with  $10^6$  nodes, for each node we store 4 real numbers: the velocity in  $x$ -,  $y$ -, and  $z$ -direction, plus the pressure;
  - we need 1,000 time steps to reach steady-state;
  - we have 10 different design parameters, and want to test 10 values for each of them  $\rightsquigarrow 10^{10}$  configurations, i.e.,  $10^{10}$  transient simulations.
- Assuming you have a HPC cluster allowing you to do so in reasonable time, just storing all trajectories (using **floats**) requires

$$10^{10} \cdot 10^3 \cdot 10^6 \cdot 4 \cdot 4 = 1.6 \cdot 10^{20} \text{ bytes} \approx 160 \text{ exabytes of memory!}$$

Data can be compressed, but first it needs to be generated and stored...



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**
  - we use a grid with  $10^6$  nodes, for each node we store 4 real numbers: the velocity in x-, y-, and z-direction, plus the pressure;
  - we need 1,000 time steps to reach steady-state;
  - we have 10 different design parameters, and want to test 10 values for each of them  $\rightsquigarrow 10^{10}$  configurations, i.e.,  $10^{10}$  transient simulations.
- Assuming you have a HPC cluster allowing you to do so in reasonable time, just storing all trajectories (using **floats**) requires
$$10^{10} \cdot 10^3 \cdot 10^6 \cdot 4 \cdot 4 = 1.6 \cdot 10^{20} \text{ bytes} \approx 160 \text{ exabytes of memory!}$$
Data can be compressed, but first it needs to be generated and stored. . .
- Why not starting with a compressed data format from the very beginning, and computing all trajectories **all-at-once**?



## A (big) data problem in flow simulation

- Assume you want to design an airfoil, wing, car, etc.
- This requires 3D flow simulations for varying configurations, and you want to save the data for visualization and further studies/comparisons.
- **Gedankenexperiment:**
  - we use a grid with  $10^6$  nodes, for each node we store 4 real numbers: the velocity in x-, y-, and z-direction, plus the pressure;
  - we need 1,000 time steps to reach steady-state;
  - we have 10 different design parameters, and want to test 10 values for each of them  $\rightsquigarrow 10^{10}$  configurations, i.e.,  $10^{10}$  transient simulations.
- Assuming you have a HPC cluster allowing you to do so in reasonable time, just storing all trajectories (using **floats**) requires
$$10^{10} \cdot 10^3 \cdot 10^6 \cdot 4 \cdot 4 = 1.6 \cdot 10^{20} \text{ bytes} \approx 160 \text{ exabytes of memory!}$$
Data can be compressed, but first it needs to be generated and stored. . .
- Why not starting with a compressed data format from the very beginning, and computing all trajectories **all-at-once**?
- This is the idea we pursue in this talk, where we store the data in a compressed (low-rank)  $1 + 10 + 1(3)$ -way tensor! (Above example  $\rightsquigarrow$  terabyte-range.)



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.
- PDEs governing the processes can involve uncertain coefficients, or uncertain sources, or uncertain geometry.



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.
- PDEs governing the processes can involve uncertain coefficients, or uncertain sources, or uncertain geometry.
- Uncertain parameters modeled as random variables  $\rightsquigarrow$  **random PDEs**, potentially also containing uncertain inputs (controls)  $\rightsquigarrow$  (generalized) polynomial chaos approach  $\rightsquigarrow$  **high-dimensional PDE!**



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.
- PDEs governing the processes can involve uncertain coefficients, or uncertain sources, or uncertain geometry.
- Uncertain parameters modeled as random variables  $\rightsquigarrow$  **random PDEs**, potentially also containing uncertain inputs (controls)  $\rightsquigarrow$  (generalized) polynomial chaos approach  $\rightsquigarrow$  **high-dimensional PDE!**
- Here: no stochastic PDEs in the sense of dynamics driven by Wiener or Lévy or ... processes!



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.
- PDEs governing the processes can involve uncertain coefficients, or uncertain sources, or uncertain geometry.
- Uncertain parameters modeled as random variables  $\rightsquigarrow$  **random PDEs**, potentially also containing uncertain inputs (controls)  $\rightsquigarrow$  (generalized) polynomial chaos approach  $\rightsquigarrow$  **high-dimensional PDE!**
- Here: no stochastic PDEs in the sense of dynamics driven by Wiener or Lévy or ... processes!

## Uncertainty arises because

- available data are incomplete;



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.
- PDEs governing the processes can involve uncertain coefficients, or uncertain sources, or uncertain geometry.
- Uncertain parameters modeled as random variables  $\rightsquigarrow$  **random PDEs**, potentially also containing uncertain inputs (controls)  $\rightsquigarrow$  (generalized) polynomial chaos approach  $\rightsquigarrow$  **high-dimensional PDE!**
- Here: no stochastic PDEs in the sense of dynamics driven by Wiener or Lévy or ... processes!

## Uncertainty arises because

- available data are incomplete;
- data are predictable, but difficult to measure, e.g., porosity above oil reservoirs;



## PDEs with stochastic coefficients

- Physical, biological, chemical, etc. processes involve uncertainties.
- Models of these processes should account for these uncertainties.
- PDEs governing the processes can involve uncertain coefficients, or uncertain sources, or uncertain geometry.
- Uncertain parameters modeled as random variables  $\rightsquigarrow$  **random PDEs**, potentially also containing uncertain inputs (controls)  $\rightsquigarrow$  (generalized) polynomial chaos approach  $\rightsquigarrow$  **high-dimensional PDE!**
- Here: no stochastic PDEs in the sense of dynamics driven by Wiener or Lévy or ... processes!

## Uncertainty arises because

- available data are incomplete;
- data are predictable, but difficult to measure, e.g., porosity above oil reservoirs;
- data are unpredictable, e.g., wind shear.



## Curse of Dimensionality

[BELLMAN '57]

Increase in matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\rightsquigarrow$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).



## Curse of Dimensionality

[BELLMAN '57]

Increase in matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\leadsto$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

Consider  $-\Delta u = f$  in  $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ , uniformly discretized as

$$(I \otimes A + A \otimes I)x =: \mathcal{A}x = b \quad \Longleftrightarrow \quad AX + XA^T = B$$

with  $x = \text{vec}(X)$  and  $b = \text{vec}(B)$  with low-rank right hand side  $B \approx b_1 b_2^T$ .



## Curse of Dimensionality

[BELLMAN '57]

Increase in matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\rightsquigarrow$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

Consider  $-\Delta u = f$  in  $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ , uniformly discretized as

$$(I \otimes A + A \otimes I)x =: \mathcal{A}x = b \quad \Longleftrightarrow \quad AX + XA^T = B$$

with  $x = \text{vec}(X)$  and  $b = \text{vec}(B)$  with low-rank right hand side  $B \approx b_1 b_2^T$ .

- Low-rankness of  $\tilde{X} := VW^T \approx X$  follows from properties of  $A$  and  $B$ , and in particular **(approximate) separability**  $u(x, y) \approx v(x)w(y)$ ,  $f(x, y) \approx g(x)h(y)$ ; e.g., [PENZL '00, GRASEDYCK '04].



## Curse of Dimensionality

[BELLMAN '57]

Increase in matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\rightsquigarrow$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

Consider  $-\Delta u = f$  in  $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ , uniformly discretized as

$$(I \otimes A + A \otimes I)x =: \mathcal{A}x = b \quad \Longleftrightarrow \quad AX + XA^T = B$$

with  $x = \text{vec}(X)$  and  $b = \text{vec}(B)$  with low-rank right hand side  $B \approx b_1 b_2^T$ .

- Low-rankness of  $\tilde{X} := VW^T \approx X$  follows from properties of  $A$  and  $B$ , and in particular (approximate) separability  $u(x, y) \approx v(x)w(y)$ ,  $f(x, y) \approx g(x)h(y)$ ; e.g., [PENZL '00, GRASEDYCK '04].
- We solve this using low-rank Krylov subspace solvers. These essentially require matrix-vector multiplication and vector computations.



## Curse of Dimensionality

[BELLMAN '57]

Increase in matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\leadsto$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

Consider  $-\Delta u = f$  in  $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ , uniformly discretized as

$$(I \otimes A + A \otimes I)x =: \mathcal{A}x = b \quad \Longleftrightarrow \quad AX + XA^T = B$$

with  $x = \text{vec}(X)$  and  $b = \text{vec}(B)$  with low-rank right hand side  $B \approx b_1 b_2^T$ .

- Low-rankness of  $\tilde{X} := VW^T \approx X$  follows from properties of  $A$  and  $B$ , and in particular (approximate) separability  $u(x, y) \approx v(x)w(y)$ ,  $f(x, y) \approx g(x)h(y)$ ; e.g., [PENZL '00, GRASEDYCK '04].
- We solve this using low-rank Krylov subspace solvers. These essentially require matrix-vector multiplication and vector computations.
- Hence,  $\mathcal{A}\text{vec}(X_k) = \mathcal{A}\text{vec}(V_k W_k^T) = \text{vec}\left([AV_k, V_k][W_k, AW_k]^T\right)$



## Curse of Dimensionality

[BELLMAN '57]

Increase in matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\rightsquigarrow$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

Consider  $-\Delta u = f$  in  $[0, 1] \times [0, 1] \subset \mathbb{R}^2$ , uniformly discretized as

$$(I \otimes A + A \otimes I)x =: \mathcal{A}x = b \quad \Longleftrightarrow \quad AX + XA^T = B$$

with  $x = \text{vec}(X)$  and  $b = \text{vec}(B)$  with low-rank right hand side  $B \approx b_1 b_2^T$ .

- Low-rankness of  $\tilde{X} := VW^T \approx X$  follows from properties of  $A$  and  $B$ , and in particular (approximate) separability  $u(x, y) \approx v(x)w(y)$ ,  $f(x, y) \approx g(x)h(y)$ ; e.g., [PENZL '00, GRASEDYCK '04].
- We solve this using low-rank Krylov subspace solvers. These essentially require matrix-vector multiplication and vector computations.
- Hence,  $\mathcal{A}\text{vec}(X_k) = \mathcal{A}\text{vec}(V_k W_k^T) = \text{vec}\left([AV_k, V_k][W_k, AW_k]^T\right)$
- The rank of  $[AV_k, V_k] \in \mathbb{R}^{n, 2r}$ ,  $[W_k, AW_k] \in \mathbb{R}^{n_t, 2r}$  increases but can be controlled using truncation.  $\rightsquigarrow$  **Low-rank Krylov subspace solvers**.

[KRESSNER/TOBLER, B/BREITEN, SAVOSTYANOV/DOLGOV, ...].



We consider the problem:

$$\min_{y \in \mathcal{Y}, u \in \mathcal{U}} \mathcal{J}(y, u) \quad \text{subject to} \quad c(y, u) = 0,$$

where

- $c(y, u) = 0$  represents a (linear or nonlinear) PDE (system) with uncertain coefficient(s).
- The state  $y$  and control  $u$  are random fields.
- The cost functional  $\mathcal{J}$  is a real-valued Fréchet-differentiable functional on  $\mathcal{Y} \times \mathcal{U}$ .



## Curse of Dimensionality

[BELLMAN '57]

Increase matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\rightsquigarrow$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

## Goal of this talk

Apply low-rank iterative solvers to discrete optimality systems resulting from

**PDE-constrained optimization problems under uncertainty,**

and go one step further applying low-rank tensor (instead of matrix) techniques.



## Curse of Dimensionality

[BELLMAN '57]

Increase matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

⇒ **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

## Goal of this talk

Apply low-rank iterative solvers to discrete optimality systems resulting from

**PDE-constrained optimization problems under uncertainty,**

and go one step further applying low-rank tensor (instead of matrix) techniques.

## Take home message

Biggest problem solved so far has  $n = 1.29 \cdot 10^{15}$  unknowns (KKT system for unsteady incompressible Navier-Stokes control problem with uncertain viscosity).



## Curse of Dimensionality

[BELLMAN '57]

Increase matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

$\rightsquigarrow$  **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

## Goal of this talk

Apply low-rank iterative solvers to discrete optimality systems resulting from

**PDE-constrained optimization problems under uncertainty,**

and go one step further applying low-rank tensor (instead of matrix) techniques.

## Take home message

Biggest problem solved so far has  $n = 1.29 \cdot 10^{15}$  unknowns (KKT system for unsteady incompressible Navier-Stokes control problem with uncertain viscosity).

Would require  $\approx 10$  **petabytes (PB)** = 10,000 **TB** to store the solution vector!



## Curse of Dimensionality

[BELLMAN '57]

Increase matrix size of discretized differential operator for  $h \rightarrow \frac{h}{2}$  by factor  $2^d$ .

↪ **Rapid Increase of Dimensionality**, called **Curse of Dimensionality** ( $d > 3$ ).

## Goal of this talk

Apply low-rank iterative solvers to discrete optimality systems resulting from

**PDE-constrained optimization problems under uncertainty,**

and go one step further applying low-rank tensor (instead of matrix) techniques.

## Take home message

Biggest problem solved so far has  $n = 1.29 \cdot 10^{15}$  unknowns (KKT system for unsteady incompressible Navier-Stokes control problem with uncertain viscosity).

Would require  $\approx 10$  **petabytes (PB)** = 10,000 **TB** to store the solution vector!

Using low-rank tensor techniques, we need  $\approx 7 \cdot 10^7$  **bytes** = 70 **GB** to solve the KKT system in MATLAB in less than one hour!



## 1. Introduction

## 2. Unsteady Heat Equation

- Model set-up

- The stochastic Galerkin finite element method

- The tensor train format

- Numerical results

## 3. Unsteady Navier-Stokes Equations

## 4. Conclusions

Consider the optimization problem

$$\mathcal{J}(t, y, u) = \frac{1}{2} \|y - \bar{y}\|_{L^2(0, T; \mathcal{D}) \otimes L^2(\Omega)}^2 + \frac{\alpha}{2} \|\text{std}(y)\|_{L^2(0, T; \mathcal{D})}^2 + \frac{\beta}{2} \|u\|_{L^2(0, T; \mathcal{D}) \otimes L^2(\Omega)}^2$$

subject,  $\mathbb{P}$ -almost surely, to

$$\begin{cases} \frac{\partial y(t, \mathbf{x}, \omega)}{\partial t} - \nabla \cdot (a(\mathbf{x}, \omega) \nabla y(t, \mathbf{x}, \omega)) = u(t, \mathbf{x}, \omega), & \text{in } (0, T] \times \mathcal{D} \times \Omega, \\ y(t, \mathbf{x}, \omega) = 0, & \text{on } (0, T] \times \partial \mathcal{D} \times \Omega, \\ y(0, \mathbf{x}, \omega) = y_0, & \text{in } \mathcal{D} \times \Omega, \end{cases}$$

where

- for any  $z : \mathcal{D} \times \Omega \rightarrow \mathbb{R}$ ,  $z(\mathbf{x}, \cdot)$  is a random variable defined on the complete probability space  $(\Omega, \mathcal{F}, \mathbb{P})$  for each  $\mathbf{x} \in \mathcal{D}$ ,
- $\exists 0 < a_{\min} < a_{\max} < \infty$  s.t.  $\mathbb{P}(\omega \in \Omega : a(\mathbf{x}, \omega) \in [a_{\min}, a_{\max}] \forall \mathbf{x} \in \mathcal{D}) = 1$ .



We discretize and then optimize the stochastic control problem.

- Under finite noise assumption we can use  $N$ -term (truncated) **Karhunen-Loève expansion (KLE)**

$$a \equiv a(\mathbf{x}, \omega) \approx a_N(\mathbf{x}, \xi(\omega)) \equiv a_N(\mathbf{x}, \xi_1(\omega), \xi_2(\omega), \dots, \xi_N(\omega)).$$

- Assuming a known continuous covariance  $C_a(\mathbf{x}, \mathbf{y})$ , we get the KLE

$$a_N(\mathbf{x}, \xi(\omega)) = \mathbb{E}[a](\mathbf{x}) + \sigma_a \sum_{i=1}^N \sqrt{\lambda_i} \varphi_i(\mathbf{x}) \xi_i(\omega),$$

where  $(\lambda_i, \varphi_i)$  are the dominant eigenpairs of  $C_a$ .

- Doob-Dynkin Lemma allows same parametrization for solution  $y$ .
- Use linear finite elements for the spatial discretization and implicit Euler in time.

This is used within a **stochastic Galerkin FEM (SGFEM)** approach.



### Weak formulation of the random PDE

Seek  $y \in H^1(0, T; H_0^1(\mathcal{D}) \otimes L^2(\Omega))$  such that,  $\mathbb{P}$ -almost surely,

$$\langle y_t, v \rangle + \mathcal{B}(y, v) = \ell(u, v) \quad \forall v \in H_0^1(\mathcal{D}) \otimes L^2(\Omega),$$

with the coercive (as  $\mathbb{P}(a(\mathbf{x}, \omega) \geq a_{\min} > 0) = 1$ ) bilinear form

$$\mathcal{B}(y, v) := \int_{\Omega} \int_{\mathcal{D}} a(\mathbf{x}, \omega) \nabla y(\mathbf{x}, \omega) \cdot \nabla v(\mathbf{x}, \omega) d\mathbf{x} d\mathbb{P}(\omega), \quad v, y \in H_0^1(\mathcal{D}) \otimes L^2(\Omega),$$

and

$$\begin{aligned} \ell(u, v) &= \langle u(\mathbf{x}, \omega), v(\mathbf{x}, \omega) \rangle \\ &=: \int_{\Omega} \int_{\mathcal{D}} u(\mathbf{x}, \omega) v(\mathbf{x}, \omega) d\mathbf{x} d\mathbb{P}(\omega), \quad u, v \in H_0^1(\mathcal{D}) \otimes L^2(\Omega). \end{aligned}$$

Coercivity and boundedness of  $\mathcal{B}$  + Lax-Milgram  $\implies$  **unique solution exists.**

---

[DEB/BABUŠKA/ODEN '01, XIU/KARNIADAKIS '03, BABUŠKA/TEMPONE/ZOURARIS '04, ROSSEEL/WELLS '12, GUNZBURGER/WEBSTER/ZHANG '14]



## Weak formulation of the optimality system

## Theorem

[CHEN/QUARTERONI '14, B./ONWUNTA/STOLL '18]

Under appropriate regularity assumptions, there exists a unique **adjoint state**  $p$  and optimal solution  $(y, u, p)$  to the optimal control problem for the random unsteady heat equation, satisfying the **stochastic optimality conditions (KKT system)** for  $t \in (0, T]$   $\mathbb{P}$ -almost surely

$$\begin{aligned}\langle y_t, v \rangle + \mathcal{B}(y, v) &= \ell(u, v), & \forall v \in H_0^1(\mathcal{D}) \otimes L^2(\Omega), \\ \langle p_t, w \rangle - \mathcal{B}^*(p, w) &= \ell \left( (y - \bar{y}) + \frac{\alpha}{2} \mathcal{S}(y), w \right), & \forall w \in H_0^1(\mathcal{D}) \otimes L^2(\Omega), \\ \ell(\beta u - p, \tilde{w}) &= 0, & \forall \tilde{w} \in L^2(\mathcal{D}) \otimes L^2(\Omega),\end{aligned}$$

where

- $\mathcal{S}(y)$  is the Fréchet derivative of  $\|\text{std}(y)\|_{L^2(0, T; \mathcal{D})}^2$ ;
- $\mathcal{B}^*$  is the adjoint operator of  $\mathcal{B}$ .



## Discretization of the random PDE

- $y, p, u$  are approximated using standard Galerkin ansatz, yielding approximations of the form

$$z(t, \mathbf{x}, \omega) = \sum_{k=0}^{P-1} \sum_{j=1}^J z_{jk}(t) \phi_j(\mathbf{x}) \psi_k(\xi) = \sum_{k=0}^{P-1} z_k(t, \mathbf{x}) \psi_k(\xi).$$

- Here,
  - $\{\phi_j\}_{j=1}^J$  are linear finite elements;
  - $\{\psi_k\}_{k=0}^{P-1}$  are the  $P = \frac{(N+n)!}{N!n!}$  multivariate Legendre polynomials of degree  $\leq n$ .
  - Implicit Euler/dG(0) used for temporal discretization with constant time step  $\tau$ .



## Discrete first order optimality conditions/KKT system

$$\begin{bmatrix} \tau \mathcal{M}_1 & 0 & -\mathcal{K}_t^T \\ 0 & \beta \tau \mathcal{M}_2 & \tau \mathcal{N}^T \\ -\mathcal{K}_t & \tau \mathcal{N} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau \mathcal{M}_\alpha \bar{\mathbf{y}} \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix},$$

where

- $\mathcal{M}_1 = D \otimes G_\alpha \otimes M =: D \otimes \mathcal{M}_\alpha, \quad \mathcal{M}_2 = D \otimes G_0 \otimes M,$
- $\mathcal{K}_t = I_{n_t} \otimes \left[ \sum_{i=0}^N G_i \otimes \hat{K}_i \right] + (C \otimes G_0 \otimes M),$
- $\mathcal{N} = I_{n_t} \otimes G_0 \otimes M,$

and

- $G_0 = \text{diag}(\langle \psi_0^2 \rangle, \langle \psi_1^2 \rangle, \dots, \langle \psi_{P-1}^2 \rangle), \quad G_i(j, k) = \langle \xi_i \psi_j \psi_k \rangle, \quad i = 1, \dots, N,$
- $G_\alpha = G_0 + \alpha \text{diag}(0, \langle \psi_1^2 \rangle, \dots, \langle \psi_{P-1}^2 \rangle) \quad (\text{with first moments } \langle \cdot \rangle \text{ w.r.t. } \mathbb{P}),$
- $\hat{K}_0 = M + \tau K_0, \quad \hat{K}_i = \tau K_i, \quad i = 1, \dots, N,$
- $M, K_i \in \mathbb{R}^{J \times J}$  are the mass and stiffness matrices w.r.t. the spatial discretization, where  $K_i$  corresponds to the contributions of the  $i$ th KLE term to the stiffness,
- $C = -\text{diag}(\text{ones}, -1), \quad D = \text{diag}(\frac{1}{2}, 1, \dots, 1, \frac{1}{2}) \in \mathbb{R}^{n_t \times n_t}.$



Discrete first order optimality conditions/KKT system

$$\begin{bmatrix} \tau \mathcal{M}_1 & 0 & -\mathcal{K}_t^T \\ 0 & \beta \tau \mathcal{M}_2 & \tau \mathcal{N}^T \\ -\mathcal{K}_t & \tau \mathcal{N} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \tau \mathcal{M}_\alpha \bar{\mathbf{y}} \\ \mathbf{0} \\ \mathbf{d} \end{bmatrix},$$

**Linear system with  $3JP_{n_t}$  unknowns!**



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \cdot$$

- Very large scale setting, (block-)structured sparsity  $\rightsquigarrow$  iterative solution.



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \cdot$$

- Very large scale setting, (block-)structured sparsity  $\rightsquigarrow$  iterative solution.
- Krylov subspace methods for indefinite symmetric systems: **MINRES**, . . . .



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}.$$

- Very large scale setting, (block-)structured sparsity  $\rightsquigarrow$  iterative solution.
- Krylov subspace methods for indefinite symmetric systems: **MINRES**, . . . .
- Requires good preconditioner.



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}.$$

- Very large scale setting, (block-)structured sparsity  $\rightsquigarrow$  iterative solution.
- Krylov subspace methods for indefinite symmetric systems: **MINRES**, . . . .
- Requires good preconditioner.
- Famous three-iterations-convergence result [MURPHY/GOLUB/WATHEN '00]: using ideal preconditioner

$$\mathcal{P} := \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \quad \text{with the Schur complement} \quad S := -BA^{-1}B^T,$$

MINRES finds the exact solution in at most three steps.



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}.$$

- Very large scale setting, (block-)structured sparsity  $\rightsquigarrow$  iterative solution.
- Krylov subspace methods for indefinite symmetric systems: **MINRES**, . . . .
- Requires good preconditioner.
- Famous three-iterations-convergence result [MURPHY/GOLUB/WATHEN '00]: using ideal preconditioner

$$\mathcal{P} := \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \quad \text{with the Schur complement} \quad S := -BA^{-1}B^T,$$

MINRES finds the exact solution in at most three steps.

- Motivates to use **approximate Schur complement preconditioner**  $\begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}.$



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}.$$

- Very large scale setting, (block-)structured sparsity  $\rightsquigarrow$  iterative solution.
- Krylov subspace methods for indefinite symmetric systems: **MINRES**, . . . .
- Requires good preconditioner.
- Famous three-iterations-convergence result [MURPHY/GOLUB/WATHEN '00]: using ideal preconditioner

$$\mathcal{P} := \begin{bmatrix} A & 0 \\ 0 & S \end{bmatrix} \quad \text{with the Schur complement} \quad S := -BA^{-1}B^T,$$

MINRES finds the exact solution in at most three steps.

- Motivates to use **approximate Schur complement preconditioner**  $\begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}$ .
- Here,  $A \sim$  mass matrices  $\rightsquigarrow$  application of  $A^{-1}$  is approximated using a small number of Chebyshev semi-iterations.



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \quad \text{with approximate Schur complement preconditioner} \quad \begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}.$$

- How to approximate the application of the inverse Schur complement efficiently?



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \quad \text{with approximate Schur complement preconditioner} \quad \begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}.$$

- How to approximate the application of the inverse Schur complement efficiently?
- Pioneering work by **BIROS, ELMAN, ERNST, GHATTAS, POWELL, SILVESTER, ULLMANN, ...**



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \quad \text{with approximate Schur complement preconditioner} \quad \begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}.$$

- How to approximate the application of the inverse Schur complement efficiently?
- Pioneering work by **BIROS, ELMAN, ERNST, GHATTAS, POWELL, SILVESTER, ULLMANN, ...**

### Theorem

[B./ONWUNTA/STOLL '16]

Let  $\alpha \in [0, +\infty)$ ,  $\gamma = \sqrt{(1+\alpha)/\beta}$ ,  $\mathcal{K} = \sum_{i=0}^N G_i \otimes K_i$ , and

$$\tilde{S} = \frac{1}{\tau} (\mathcal{K} + \tau\gamma\mathcal{N}) \mathcal{M}_1^{-1} (\mathcal{K} + \tau\gamma\mathcal{N})^T.$$

Then the eigenvalues of  $\tilde{S}^{-1}S$  satisfy

$$\lambda(\tilde{S}^{-1}S) \subset \left[ \frac{1}{2(1+\alpha)}, 1 \right), \quad \forall \alpha < \left( \frac{\sqrt{\kappa(\mathcal{K})} + 1}{\sqrt{\kappa(\mathcal{K})} - 1} \right)^2 - 1.$$



Optimality system leads to saddle point problem

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \quad \text{with approximate Schur complement preconditioner} \quad \begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}.$$

- How to approximate the application of the inverse Schur complement efficiently?
- Pioneering work by BIROS, ELMAN, ERNST, GHATTAS, POWELL, SILVESTER, ULLMANN, ...

## Corollary

[B./ONWUNTA/STOLL '16]

Let  $\mathcal{A}$  be the KKT matrix from the stochastic Galerkin approach, and  $\mathcal{P}$  the preconditioner using the Schur complement approximation  $\tilde{S}$  (and exact  $A$ ). Then

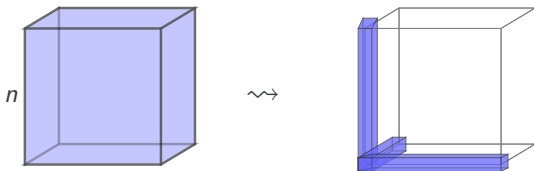
$$\lambda(\mathcal{P}^{-1}\mathcal{A}) \subset \{1\} \cup \mathcal{I}^+ \cup \mathcal{I}^-,$$

where

$$\mathcal{I}^\pm = \frac{1}{2} \left( 1 \pm \left[ \sqrt{1 + \frac{2}{1 + \alpha}}, \sqrt{5} \right] \right).$$



## Separation of variables and low-rank approximation



- Approximate:  $\underbrace{\mathbf{x}(i_1, \dots, i_d)}_{\text{tensor}} \approx \underbrace{\sum_{\alpha} \mathbf{x}_{\alpha}^{(1)}(i_1) \mathbf{x}_{\alpha}^{(2)}(i_2) \cdots \mathbf{x}_{\alpha}^{(d)}(i_d)}_{\text{tensor product decomposition}}.$

Goals:

- Store and manipulate  $\mathbf{x}$
- Solve equations  $A\mathbf{x} = \mathbf{b}$

$\mathcal{O}(dn)$  cost instead of  $\mathcal{O}(n^d)$ .

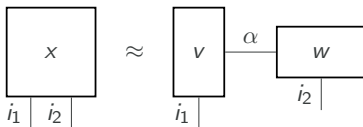
$\mathcal{O}(dn^2)$  cost instead of  $\mathcal{O}(n^{2d})$ .



- Discrete *separation of variables*:

$$\begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & & \vdots \\ x_{n,1} & \cdots & x_{n,n} \end{bmatrix} = \sum_{\alpha=1}^r \begin{bmatrix} v_{1,\alpha} \\ \vdots \\ v_{n,\alpha} \end{bmatrix} \begin{bmatrix} w_{\alpha,1} & \cdots & w_{\alpha,n} \end{bmatrix} + \mathcal{O}(\varepsilon).$$

- Diagrams:



- Rank  $r \ll n$ .
- $\text{mem}(v) + \text{mem}(w) = 2nr \ll n^2 = \text{mem}(x)$ .
- Singular Value Decomposition (SVD)  
 $\implies \varepsilon(r)$  optimal w.r.t. spectral/Frobenius norm.



## Tensor Trains/Matrix Product States

[WILSON '75, WHITE '93, VERSTRAETE '04, OSELEDETS '09/'11]

For indices

$$\overline{i_p \dots i_q} = (i_p - 1)n_{p+1} \dots n_q + (i_{p+1} - 1)n_{p+2} \dots n_q + \dots + (i_{q-1} - 1)n_q + i_q,$$

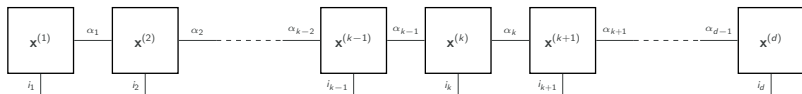
the TT format can be expressed as

$$x(\overline{i_1 \dots i_d}) = \sum_{\alpha=1}^r \mathbf{x}_{\alpha_1}^{(1)}(i_1) \cdot \mathbf{x}_{\alpha_1, \alpha_2}^{(2)}(i_2) \cdot \mathbf{x}_{\alpha_2, \alpha_3}^{(3)}(i_3) \dots \mathbf{x}_{\alpha_{d-1}, \alpha_d}^{(d)}(i_d)$$

or

$$x(\overline{i_1 \dots i_d}) = \mathbf{x}^{(1)}(i_1) \dots \mathbf{x}^{(d)}(i_d), \quad \mathbf{x}^{(k)}(i_k) \in \mathbb{R}^{r_{k-1} \times r_k} \text{ w/ } r_0, r_d = 1,$$

or

Storage:  $\mathcal{O}(dnr^2)$  instead of  $\mathcal{O}(n^d)$ .



Always work with *factors*  $\mathbf{x}^{(k)} \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$  instead of *full tensors*.

- Sum  $z = x + y \rightsquigarrow$  increase of tensor rank  $r_z = r_x + r_y$ .
- TT format for a high-dimensional operator

$$A(\overline{i_1 \dots i_d}, \overline{j_1 \dots j_d}) = \mathbf{A}^{(1)}(i_1, j_1) \cdots \mathbf{A}^{(d)}(i_d, j_d)$$

- *Matrix-vector* multiplication  $y = Ax$ ;  $\rightsquigarrow$  tensor rank  $r_y = r_A \cdot r_x$ .
- Additions and multiplications *increase* TT ranks.
- *Decrease* ranks quasi-optimally via QR and SVD.



The dimensionality of the saddle point system is vast  $\Rightarrow$  use **tensor structure** and low tensor ranks.



The dimensionality of the saddle point system is vast  $\Rightarrow$  use **tensor structure** and low tensor ranks.

Use tensor train format to approximate the solution as

$$\mathbf{y}(i_1, \dots, i_d) \approx \sum_{\alpha_1 \dots \alpha_{d-1}=1}^{r_1 \dots r_{d-1}} \mathbf{y}_{\alpha_1}^{(1)}(i_1) \mathbf{y}_{\alpha_1, \alpha_2}^{(2)}(i_2) \cdots \mathbf{y}_{\alpha_{d-2}, \alpha_{d-1}}^{(d-1)}(i_{d-1}) \mathbf{y}_{\alpha_{d-1}}^{(d)}(i_d),$$

and represent the coefficient matrix as

$$\mathcal{A}(i_1 \cdots i_d, j_1 \cdots j_d) \approx \sum_{\beta_1 \dots \beta_{d-1}=1}^{R_1 \dots R_{d-1}} \mathbf{A}_{\beta_1}^{(1)}(i_1, j_1) \mathbf{A}_{\beta_1, \beta_2}^{(2)}(i_2, j_2) \cdots \mathbf{A}_{\beta_{d-1}}^{(d)}(i_d, j_d),$$

where the multi-index  $\mathbf{i} = (i_1, \dots, i_d)$  is implied by the parametrization of the approximate solutions of the form

$$\mathbf{z}(t, \xi_1, \dots, \xi_N, \mathbf{x}), \quad \mathbf{z} = \mathbf{y}, \mathbf{u}, \mathbf{p},$$

i.e., **solution vectors are represented by  $d$ -way tensor with  $d = N + 2$ .**



## Mean-Based Preconditioned TT-MinRes

| TT-MINRES                           | # iter (t) | # iter (t) | # iter (t) |
|-------------------------------------|------------|------------|------------|
| $n_t$                               | $2^5$      | $2^6$      | $2^8$      |
| $\dim(\mathcal{A}) = 3JPn_t$        | 10,671,360 | 21,342,720 | 85,370,880 |
| $\alpha = 1, \text{ tol} = 10^{-3}$ |            |            |            |
| $\beta = 10^{-5}$                   | 6 (285.5)  | 6 (300.0)  | 8 (372.2)  |
| $\beta = 10^{-6}$                   | 4 (77.6)   | 4 (130.9)  | 4 (126.7)  |
| $\beta = 10^{-8}$                   | 4 (56.7)   | 4 (59.4)   | 4 (64.9)   |
| $\alpha = 0, \text{ tol} = 10^{-3}$ |            |            |            |
| $\beta = 10^{-5}$                   | 4 (207.3)  | 6 (366.5)  | 6 (229.5)  |
| $\beta = 10^{-6}$                   | 4 (153.9)  | 4 (158.3)  | 4 (172.0)  |
| $\beta = 10^{-8}$                   | 2 (35.2)   | 2 (37.8)   | 2 (40.0)   |



## 1. Introduction

## 2. Unsteady Heat Equation

## 3. Unsteady Navier-Stokes Equations

Model problem: von Kármán vortex street

Numerical discretization techniques

Alternating linear solvers

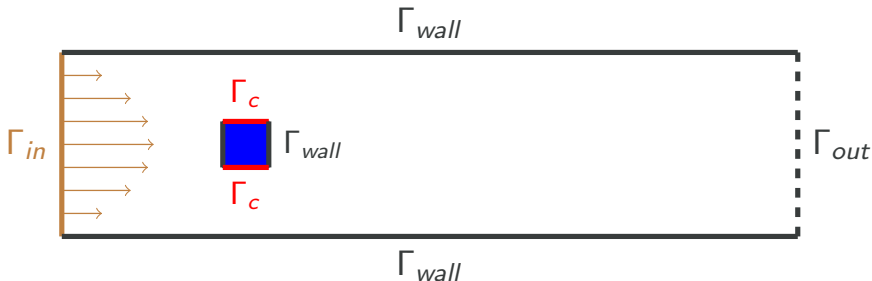
Numerical Experiments

## 4. Conclusions



Model Problem: 'Uncertain' flow past a rectangular obstacle domain

► to Numerical Experiments



- We model this as a **boundary control problem**.
- Our constraint  $c(y, u) = 0$  is given by the unsteady incompressible Navier-Stokes equations with **uncertain viscosity**  $\nu := \nu(\omega)$ .



Minimize:

$$\mathcal{J}(v, u) = \frac{1}{2} \|\operatorname{curl} v\|_{L^2(0, T; \mathcal{D}) \otimes L^2(\Omega)}^2 + \frac{\beta}{2} \|u\|_{L^2(0, T; \mathcal{D}) \otimes L^2(\Omega)}^2 \quad (1)$$

subject to

$$\begin{aligned} \frac{\partial v}{\partial t} - \nu \Delta v + (v \cdot \nabla) v + \nabla p &= 0, & \text{in } \mathcal{D}, \\ -\nabla \cdot v &= 0, & \text{in } \mathcal{D}, \\ v &= \theta, & \text{on } \Gamma_{in}, \\ v &= 0, & \text{on } \Gamma_{wall}, \\ \frac{\partial v}{\partial n} &= u, & \text{on } \Gamma_c, \\ \frac{\partial v}{\partial n} &= 0, & \text{on } \Gamma_{out}, \\ v(\cdot, 0, \cdot) &= v_0, & \text{in } \mathcal{D}. \end{aligned} \quad (2)$$



We assume

- $\nu(\omega) = \nu_0 + \nu_1 \xi(\omega)$ ,  $\nu_0, \nu_1 \in \mathbb{R}^+$ ,  $\xi \sim \mathcal{U}(-1, 1)$ .
- $\mathbb{P}(\omega \in \Omega : \nu(\omega) \in [\nu_{\min}, \nu_{\max}]) = 1$ , for some  $0 < \nu_{\min} < \nu_{\max} < +\infty$ .
- $\Rightarrow$  velocity  $v$ , control  $u$  and pressure  $p$  are random fields on  $L^2(\Omega)$ .
- $L^2(\Omega) := L^2(\Omega, \mathcal{F}, \mathbb{P})$  is a complete probability space.
- $L^2(0, T; \mathcal{D}) := L^2(\mathcal{D}) \times L^2(\mathcal{T})$ .



We assume

- $\nu(\omega) = \nu_0 + \nu_1 \xi(\omega)$ ,  $\nu_0, \nu_1 \in \mathbb{R}^+$ ,  $\xi \sim \mathcal{U}(-1, 1)$ .
- $\mathbb{P}(\omega \in \Omega : \nu(\omega) \in [\nu_{\min}, \nu_{\max}]) = 1$ , for some  $0 < \nu_{\min} < \nu_{\max} < +\infty$ .
- $\Rightarrow$  velocity  $v$ , control  $u$  and pressure  $p$  are random fields on  $L^2(\Omega)$ .
- $L^2(\Omega) := L^2(\Omega, \mathcal{F}, \mathbb{P})$  is a complete probability space.
- $L^2(0, T; \mathcal{D}) := L^2(\mathcal{D}) \times L^2(\mathcal{T})$ .

## Computational challenges

- Nonlinearity (due to the nonlinear convection term  $(v \cdot \nabla)v$ ).
- Uncertainty (due to random  $\nu(\omega)$ ).
- High dimensionality (of the resulting linear/optimality systems).

OTD Strategy and Picard (Oseen) Iteration  $\rightsquigarrow$ 

## state equation

$$\begin{aligned} v_t - \nu \Delta v + (\bar{v} \cdot \nabla) v + \nabla p &= 0 \\ \nabla \cdot v &= 0 + \text{boundary conditions} \end{aligned}$$

## adjoint equation

$$\begin{aligned} -\chi_t - \Delta \chi - (\bar{v} \cdot \nabla) \chi + (\nabla \bar{v})^T \chi + \nabla \mu &= -\text{curl}^2 v \\ \nabla \cdot \chi &= 0 \\ \text{on } \Gamma_{wall} \cup \Gamma_{in} : \quad \chi &= 0 \\ \text{on } \Gamma_{out} \cup \Gamma_c : \quad \frac{\partial \chi}{\partial n} &= 0 \\ \chi(\cdot, T, \cdot) &= 0 \end{aligned}$$

## gradient equation

$$\beta u + \chi|_{\Gamma_c} = 0.$$

OTD Strategy and Picard (Oseen) Iteration  $\rightsquigarrow$ 

## state equation

$$\begin{aligned} v_t - \nu \Delta v + (\bar{v} \cdot \nabla) v + \nabla p &= 0 \\ \nabla \cdot v &= 0 + \text{boundary conditions} \end{aligned}$$

## adjoint equation

$$\begin{aligned} -\chi_t - \Delta \chi - (\bar{v} \cdot \nabla) \chi + (\nabla \bar{v})^T \chi + \nabla \mu &= -\text{curl}^2 v \\ \nabla \cdot \chi &= 0 \\ \text{on } \Gamma_{wall} \cup \Gamma_{in} : \quad \chi &= 0 \\ \text{on } \Gamma_{out} \cup \Gamma_c : \quad \frac{\partial \chi}{\partial n} &= 0 \\ \chi(\cdot, T, \cdot) &= 0 \end{aligned}$$

## gradient equation

$$\beta u + \chi|_{\Gamma_c} = 0.$$

- $\bar{v}$  denotes the velocity from the previous Oseen iteration.
- Having solved this system, we update  $\bar{v} = v$  until convergence.



- Velocity  $v$  and control  $u$  are of the form

$$z(t, x, \omega) = \sum_{k=0}^{P-1} \sum_{j=1}^{J_v} z_{jk}(t) \phi_j(x) \psi_k(\xi) = \sum_{k=0}^{P-1} z_k(t, x) \psi_k(\xi).$$

- Pressure  $p$  is of the form

$$p(t, x, \omega) = \sum_{k=0}^{P-1} \sum_{j=1}^{J_p} p_{jk}(t) \tilde{\phi}_j(x) \psi_k(\xi) = \sum_{k=0}^{P-1} p_k(t, x) \psi_k(\xi).$$

- Here,
  - $\{\phi_j\}_{j=1}^{J_v}$  and  $\{\tilde{\phi}_j\}_{j=1}^{J_p}$  are Q2–Q1 finite elements (inf-sup stable);
  - $\{\psi_k\}_{k=0}^{P-1}$  are Legendre polynomials.
- Implicit Euler/dG(0) used for temporal discretization.



Linearization and SGFEM discretization yields the following saddle point system

$$\underbrace{\begin{bmatrix} M_y & 0 & L^* \\ 0 & M_u & N^\top \\ L & N & 0 \end{bmatrix}}_A \underbrace{\begin{bmatrix} y \\ u \\ \lambda \end{bmatrix}}_x = \underbrace{\begin{bmatrix} f \\ 0 \\ g \end{bmatrix}}_b.$$

Each of the block matrices in  $A$  is of the form

$$\sum_{\alpha=1}^R X_\alpha \otimes Y_\alpha \otimes Z_\alpha,$$

corresponding to temporal, stochastic, and spatial discretizations.



Linearization and SGFEM discretization yields the following saddle point system

$$\underbrace{\begin{bmatrix} M_y & 0 & L^* \\ 0 & M_u & N^\top \\ L & N & 0 \end{bmatrix}}_A \underbrace{\begin{bmatrix} y \\ u \\ \lambda \end{bmatrix}}_x = \underbrace{\begin{bmatrix} f \\ 0 \\ g \end{bmatrix}}_b.$$

Each of the block matrices in  $A$  is of the form

$$\sum_{\alpha=1}^R X_\alpha \otimes Y_\alpha \otimes Z_\alpha,$$

corresponding to temporal, stochastic, and spatial discretizations.

Size:  $\sim 3n_t P(J_v + J_p)$ , e.g., for  $P = 10$ ,  $n_t = 2^{10}$ ,  $J \approx 10^5 \rightsquigarrow \approx 10^9$  unknowns!



## Central Question

How to solve  $Ax = b$  if Krylov solvers become too expensive?



## Central Question

How to solve  $Ax = b$  if Krylov solvers become too expensive?

Data are given in TT format:

- $A(i, j) = \mathbf{A}^{(1)}(i_1, j_1) \cdots \mathbf{A}^{(d)}(i_d, j_d).$
- $b(i) = \mathbf{b}^{(1)}(i_1) \cdots \mathbf{b}^{(d)}(i_d).$

Seek the solution in the same format:

- $x(i) = \mathbf{x}^{(1)}(i_1) \cdots \mathbf{x}^{(d)}(i_d).$



## Central Question

How to solve  $Ax = b$  if Krylov solvers become too expensive?

Data are given in TT format:

- $A(i, j) = \mathbf{A}^{(1)}(i_1, j_1) \cdots \mathbf{A}^{(d)}(i_d, j_d).$
- $b(i) = \mathbf{b}^{(1)}(i_1) \cdots \mathbf{b}^{(d)}(i_d).$

Seek the solution in the same format:

- $x(i) = \mathbf{x}^{(1)}(i_1) \cdots \mathbf{x}^{(d)}(i_d).$

Use a new block-variant of *Alternating Least Squares* in a new block TT format to overcome difficulties with indefiniteness of KKT system matrix.



- If  $A = A^\top > 0$ : minimize  $J(x) = x^\top Ax - 2x^\top b$ .

### *Alternating Least Squares (ALS):*

- replace  $\min_x J(x)$  by iteration
- for  $k = 1, \dots, d$ ,  
    solve  $\min_{\mathbf{x}^{(k)}} J(\mathbf{x}^{(1)}(i_1) \cdots \mathbf{x}^{(k)}(i_k) \cdots \mathbf{x}^{(d)}(i_d))$ .  
    (all other blocks are fixed)

size  $n^d$

size  $r^2 n$



$$1. \hat{\mathbf{x}}^{(1)} = \arg \min_{\mathbf{x}^{(1)}} J(\mathbf{x}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$$



1.  $\hat{\mathbf{x}}^{(1)} = \arg \min_{\mathbf{x}^{(1)}} J(\mathbf{x}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
2.  $\hat{\mathbf{x}}^{(2)} = \arg \min_{\mathbf{x}^{(2)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$



1.  $\hat{\mathbf{x}}^{(1)} = \arg \min_{\mathbf{x}^{(1)}} J(\mathbf{x}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
2.  $\hat{\mathbf{x}}^{(2)} = \arg \min_{\mathbf{x}^{(2)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
3.  $\hat{\mathbf{x}}^{(3)} = \arg \min_{\mathbf{x}^{(3)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \hat{\mathbf{x}}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$



1.  $\hat{\mathbf{x}}^{(1)} = \arg \min_{\mathbf{x}^{(1)}} J(\mathbf{x}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
2.  $\hat{\mathbf{x}}^{(2)} = \arg \min_{\mathbf{x}^{(2)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
3.  $\hat{\mathbf{x}}^{(3)} = \arg \min_{\mathbf{x}^{(3)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \hat{\mathbf{x}}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
4.  $\mathbf{x}^{(2)} = \arg \min_{\mathbf{x}^{(2)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \hat{\mathbf{x}}^{(3)}(i_3))$



1.  $\hat{\mathbf{x}}^{(1)} = \arg \min_{\mathbf{x}^{(1)}} J(\mathbf{x}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
2.  $\hat{\mathbf{x}}^{(2)} = \arg \min_{\mathbf{x}^{(2)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
3.  $\hat{\mathbf{x}}^{(3)} = \arg \min_{\mathbf{x}^{(3)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \hat{\mathbf{x}}^{(2)}(i_2) \mathbf{x}^{(3)}(i_3))$
4.  $\mathbf{x}^{(2)} = \arg \min_{\mathbf{x}^{(2)}} J(\hat{\mathbf{x}}^{(1)}(i_1) \mathbf{x}^{(2)}(i_2) \hat{\mathbf{x}}^{(3)}(i_3))$
5. repeat 1.–4. until convergence



If we differentiate  $J$  w.r.t. TT blocks, we see that...

- ... each step means solving a *Galerkin linear system*

$$\left(X_{\neq k}^\top A X_{\neq k}\right) \hat{\mathbf{x}}^{(k)} = \left(X_{\neq k}^\top \mathbf{b}\right) \in \mathbb{R}^{nr^2}.$$

$$\bullet X_{\neq k} = \underbrace{\text{TT} \left( \hat{\mathbf{x}}^{(1)} \dots \hat{\mathbf{x}}^{(k-1)} \right)}_{n^{k-1} \times r_{k-1}} \otimes \underbrace{I}_{n \times n} \otimes \underbrace{\text{TT} \left( \mathbf{x}^{(k+1)} \dots \mathbf{x}^{(d)} \right)}_{n^{d-k} \times r_k}.$$



If we differentiate  $J$  w.r.t. TT blocks, we see that...

- ... each step means solving a *Galerkin linear system*

$$\left(X_{\neq k}^\top A X_{\neq k}\right) \hat{\mathbf{x}}^{(k)} = \left(X_{\neq k}^\top \mathbf{b}\right) \in \mathbb{R}^{nr^2}.$$

$$\bullet X_{\neq k} = \underbrace{\text{TT}\left(\hat{\mathbf{x}}^{(1)} \dots \hat{\mathbf{x}}^{(k-1)}\right)}_{n^{k-1} \times r_{k-1}} \otimes \underbrace{I}_{n \times n} \otimes \underbrace{\text{TT}\left(\mathbf{x}^{(k+1)} \dots \mathbf{x}^{(d)}\right)}_{n^{d-k} \times r_k}.$$

## Properties of ALS include:

- + Effectively 1D complexity in a prescribed format.
- Tensor format (ranks) is fixed and cannot be adapted.
- Convergence may be very slow, stagnation is likely.



- Density Matrix Renormalization Group (DMRG) [WHITE '92]
  - updates *two* blocks  $\mathbf{x}^{(k)}\mathbf{x}^{(k+1)}$  *simultaneously*.
- Alternating Minimal Energy (AMEn) [DOLGOV/SAVOSTYANOV '13]
  - *augments*  $X_{\neq k}$  by a TT block of the *residual*  $\mathbf{z}^{(k)}$ .



- Density Matrix Renormalization Group (DMRG) [WHITE '92]  
– updates *two* blocks  $\mathbf{x}^{(k)}\mathbf{x}^{(k+1)}$  *simultaneously*.
- Alternating Minimal Energy (AMEn) [DOLGOV/SAVOSTYANOV '13]  
– *augments*  $X_{\neq k}$  by a TT block of the *residual*  $\mathbf{z}^{(k)}$ .

But... , what about saddle point systems  $A$ ?

- Recall our KKT system:

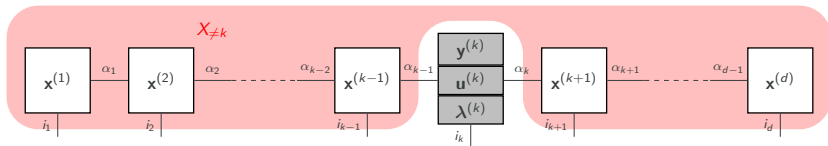
$$\underbrace{\begin{bmatrix} M_y & 0 & L^* \\ 0 & M_u & N^\top \\ L & N & 0 \end{bmatrix}}_A \begin{bmatrix} y \\ u \\ \lambda \end{bmatrix} = \begin{bmatrix} f \\ 0 \\ g \end{bmatrix}.$$

- The whole matrix is **indefinite**  $\Rightarrow X_{\neq k}^\top A X_{\neq k}$  can be degenerate.



- Work-around: Block TT representation

$$\begin{bmatrix} y \\ u \\ \lambda \end{bmatrix} = \mathbf{x}_{\alpha_1}^{(1)} \otimes \cdots \otimes \begin{bmatrix} \mathbf{y}_{\alpha_{k-1}, \alpha_k}^{(k)} \\ \mathbf{u}_{\alpha_{k-1}, \alpha_k}^{(k)} \\ \boldsymbol{\lambda}_{\alpha_{k-1}, \alpha_k}^{(k)} \end{bmatrix} \otimes \cdots \otimes \mathbf{x}_{\alpha_{d-1}}^{(d)}.$$

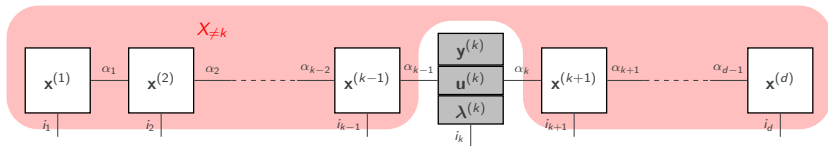


- $X_{\neq k}$  is the *same* for  $y, u, \lambda$ .



- Work-around: Block TT representation

$$\begin{bmatrix} y \\ u \\ \lambda \end{bmatrix} = \mathbf{x}_{\alpha_1}^{(1)} \otimes \cdots \otimes \begin{bmatrix} \mathbf{y}_{\alpha_{k-1}, \alpha_k}^{(k)} \\ \mathbf{u}_{\alpha_{k-1}, \alpha_k}^{(k)} \\ \lambda_{\alpha_{k-1}, \alpha_k}^{(k)} \end{bmatrix} \otimes \cdots \otimes \mathbf{x}_{\alpha_{d-1}}^{(d)}.$$



- $X_{\neq k}$  is the *same* for  $y, u, \lambda$ .
- Project each *submatrix*:

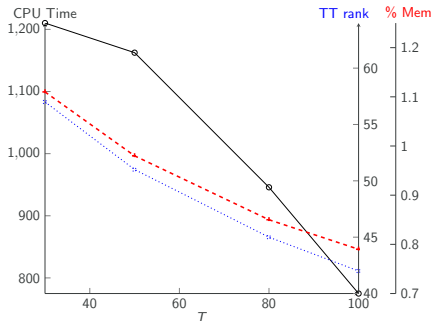
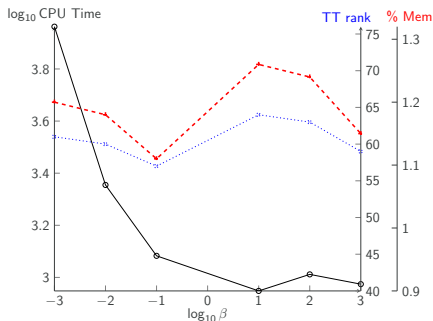
$$\begin{bmatrix} \hat{M}_y & 0 & \hat{L}^* \\ 0 & \hat{M}_u & \hat{N}^\top \\ \hat{L} & \hat{N} & 0 \end{bmatrix} \begin{bmatrix} y^{(k)} \\ u^{(k)} \\ \lambda^{(k)} \end{bmatrix} = \begin{bmatrix} \hat{f} \\ 0 \\ \hat{g} \end{bmatrix}, \quad \widehat{(\cdot)} = X_{\neq k}^\top (\cdot) X_{\neq k}$$

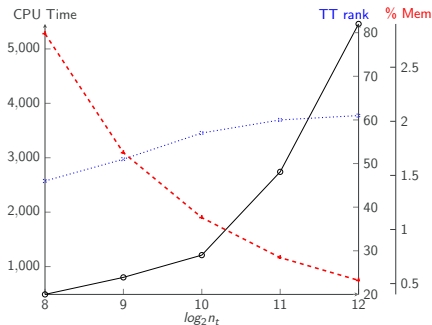
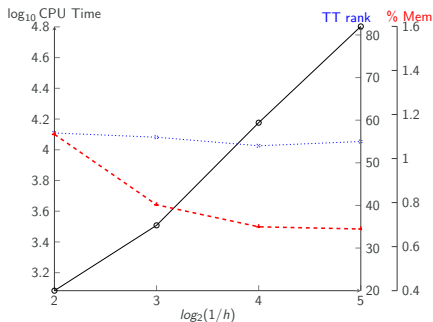


## Vary one of the default parameters:

- TT truncation tolerance  $\varepsilon = 10^{-4}$ ,
- mean viscosity  $\nu_0 = 1/20$ ,
- uncertainty  $\nu_1 = 1/80$ ,
- regularization/penalty parameter  $\beta = 10^{-1}$ ,
- number of time steps:  $n_t = 2^{10}$ ,
- time horizon  $T = 30$ ,
- spatial grid size  $h = 1/4 \rightsquigarrow J = 2488$ ,
- max. degree of Legendre polynomials:  $P = 8$ .

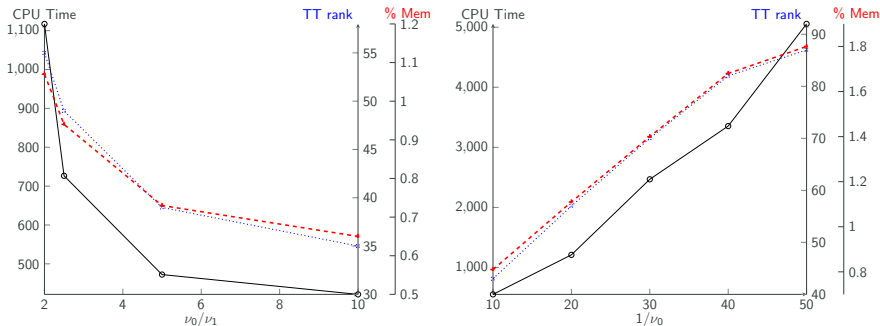
Solve projected linear systems using block-preconditioned GMRES using efficient approximation of Schur complement [B/ONWUNTA/STOLL 2016].



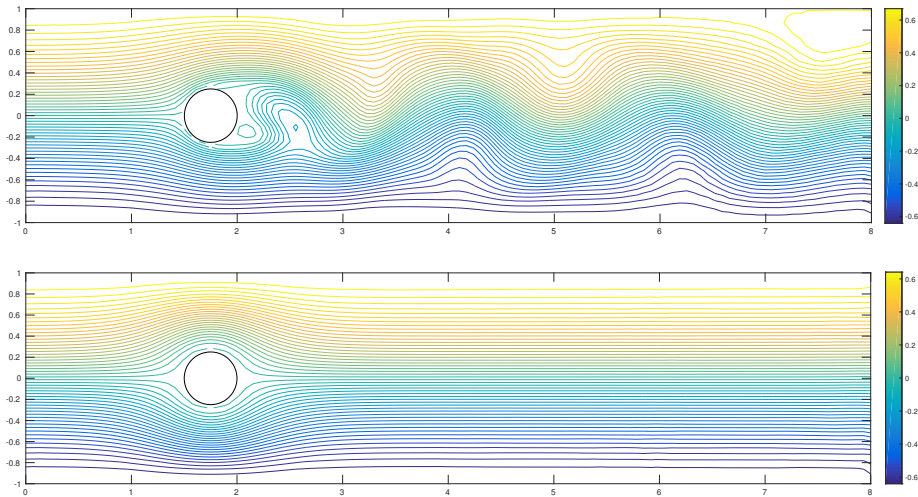


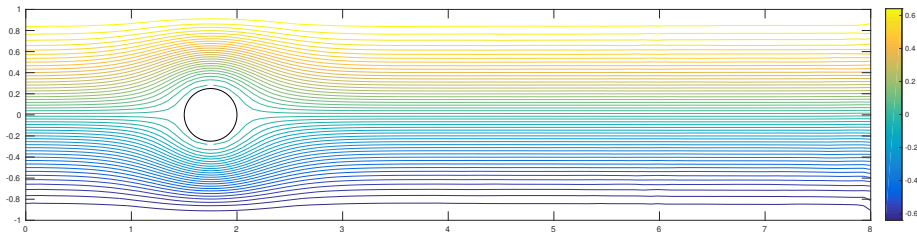
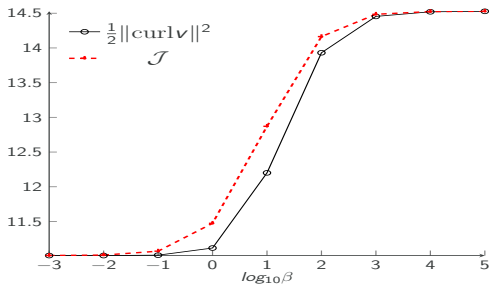


# Varying the Viscosity



**Figure:** Left:  $\nu_0 = 1/10$ ,  $\nu_1$  is varied. Right:  $\nu_1$  and  $\nu_0$  are varied together as  $\nu_1 = 0.25\nu_0$



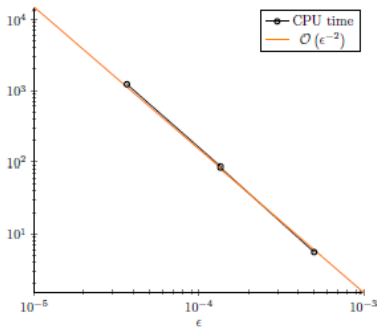




Balance all errors w.r.t.

- time, spatial, and stochastic discretization;
- TT truncation.

Call the **balanced error**  $\varepsilon$ .

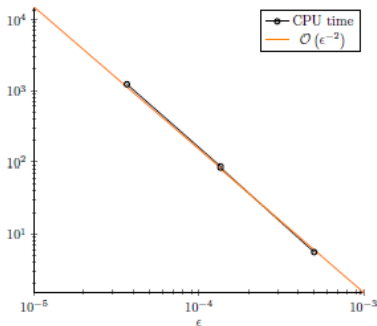




Balance all errors w.r.t.

- time, spatial, and stochastic discretization;
- TT truncation.

Call the **balanced error**  $\varepsilon$ .



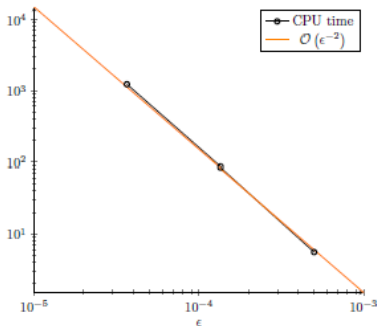
This indicates **asymptotic complexity**  $\varepsilon^{-2}$ , asymptotically equal complexity as for deterministic problem.



Balance all errors w.r.t.

- time, spatial, and stochastic discretization;
- TT truncation.

Call the **balanced error**  $\varepsilon$ .



This indicates **asymptotic complexity**  $\varepsilon^{-2}$ , asymptotically equal complexity as for deterministic problem.

This compares favorably to

- Monte-Carlo  $\mathcal{O}(\varepsilon^{-4})$ ,
- quasi Monte-Carlo / stochastic collocation  $\mathcal{O}(\varepsilon^{-3})$ .



1. Introduction
2. Unsteady Heat Equation
3. Unsteady Navier-Stokes Equations
4. Conclusions



- Low-rank tensor solver for unsteady heat and Navier-Stokes equations with uncertain viscosity.
- Similar techniques already used for ▶ 3D Stokes(-Brinkman) optimal control problems.
- With 1 stochastic parameter, the scheme reduces complexity by up to 2–3 orders of magnitude.
- To consider next:



- Low-rank tensor solver for unsteady heat and Navier-Stokes equations with uncertain viscosity.
- Similar techniques already used for ► 3D Stokes(-Brinkman) optimal control problems.
- With 1 stochastic parameter, the scheme reduces complexity by up to 2–3 orders of magnitude.
- To consider next:
  - many parameters coming from uncertain geometry or Karhunen-Loève expansion of random fields;  
Basic observation: the more parameters, the more significant is the complexity reduction w.r.t. memory — up to a factor of  $10^9$  for the control problem for a backward facing step set-up.
  - HPC implementation of AMEn-like solver to deal with even larger problems.



P. Benner, S. Dolgov, A. Onwunta, and M. Stoll.

Low-rank solvers for unsteady Stokes-Brinkman optimal control problem with random data. *COMPUTER METHODS IN APPLIED MECHANICS AND ENGINEERING*, 304:26–54, 2016.



P. Benner, A. Onwunta, and M. Stoll.

Low rank solution of unsteady diffusion equations with stochastic coefficients. *SIAM/ASA JOURNAL ON UNCERTAINTY QUANTIFICATION*, 3(1):622–649, 2015.



P. Benner, A. Onwunta, and M. Stoll.

Block-diagonal preconditioning for optimal control problems constrained by PDEs with uncertain inputs. *SIAM JOURNAL ON MATRIX ANALYSIS AND APPLICATIONS*, 37(2):491–518, 2016.



P. Benner, S. Dolgov, A. Onwunta, and M. Stoll.

Solving optimal control problems governed by random Navier-Stokes equations using low-rank methods. [arXiv:1703.06097](https://arxiv.org/abs/1703.06097), March 2017.



P. Benner, A. Onwunta, and M. Stoll.

On the existence and uniqueness of the solution of parabolic optimal control problems with uncertain inputs. *Submitted, September 2018*.



CSC

# Save the date!

## **METT-VIII**

**8th Workshop on Matrix Equations and Tensor Techniques**

**6–8 November 2019, Magdeburg, Germany**

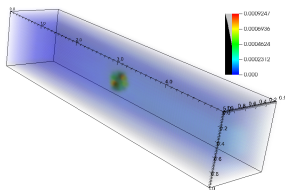
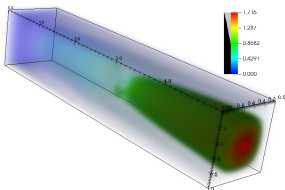
**Details coming soon!**



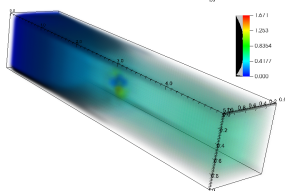
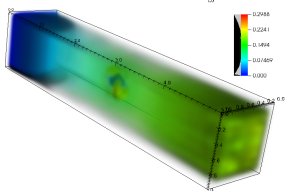
Mean

Standard deviation

State



Control



- Full size:  $n_x n_\xi n_t \approx 3 \cdot 10^9$ . Reduction:  $\frac{\text{mem}(TT)}{\text{mem}(full)} = 0.002$ .

[◀ return](#)