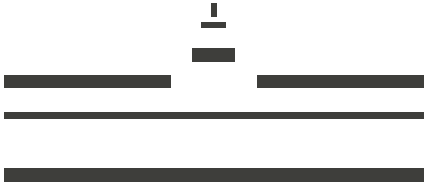


Westfälische
Wilhelms-Universität
Münster

# Model Order Reduction with pyMOR and NGSolve

Stephan Rave

living.knowledge
WWU Münster

NGSolve User Meeting

Vienna
June 15, 2017

# Outline

- Model Order Reduction.
- Model Order Reduction with pyMOR.
- Model Order Reduction with pyMOR and NGSolve.

## pyMOR main developers


Rene Milk


Petar Mlinarić


Stephan Rave


Felix Schindler

# Model Order Reduction

# Parametric Model Order Reduction

Consider parametric problems

$$\Phi : \mathcal{P} \to V, \qquad s : V \to \mathbb{R}^S$$

where

- $\mathcal{P} \subset \mathbb{R}^P$ *compact* set (parameter domain).
- $V$ Hilbert space (solution state space, dim $V \gg 0$, possibly dim $V = \infty$).
- $\Phi$ maps parameters to solutions of parametric PDE (*hard* to compute).
- $s$ maps state vectors to quantities of interest.

## Objective

Compute

$$s \circ \Phi : \mathbb{R}^P \to V \to \mathbb{R}^S$$

for *many* $\mu \in \mathcal{P}$ or *quickly* for unknown single $\mu \in \mathcal{P}$.

# The Reduced Basis Method in a Nutshell

## Objective

Quickly compute

$$s \circ \Phi \quad : \mathbb{R}^P \to V \quad \to \mathbb{R}^S.$$

- When $\Phi$, $s$ are sufficiently smooth, a quickly computable low-dimensional approximation of $s \circ \Phi$ should exist.

# The Reduced Basis Method in a Nutshell

## Objective

Quickly compute

$$s \circ \Phi_N : \mathbb{R}^P \to V_N \to \mathbb{R}^S.$$

- When $\Phi$, $s$ are sufficiently smooth, a quickly computable low-dimensional approximation of $s \circ \Phi$ should exist.
- **Idea 1:** State space reduction:
  - Define approximation $\Phi_N : \mathcal{P} \to V_N$ via Galerkin projection, $\dim V_N =: N \ll \dim V$.
  - Approximate $s \circ \Phi \approx s \circ \Phi_N$.

# The Reduced Basis Method in a Nutshell

## Objective

Quickly compute

$$s \circ \Phi_N : \mathbb{R}^P \to V_N \to \mathbb{R}^S.$$

- When $\Phi$, $s$ are sufficiently smooth, a quickly computable low-dimensional approximation of $s \circ \Phi$ should exist.
- **Idea 1:** State space reduction:
  - Define approximation $\Phi_N : \mathcal{P} \to V_N$ via Galerkin projection, $\dim V_N =: N \ll \dim V$.
  - Approximate $s \circ \Phi \approx s \circ \Phi_N$.
- **Idea 2:** $V_N \subseteq \operatorname{span}\{\Phi(\mu_1), \dots, \Phi(\mu_k)\}$.

# The Reduced Basis Method in a Nutshell

## Objective

Quickly compute

$$s \circ \Phi_N : \mathbb{R}^P \to V_N \to \mathbb{R}^S.$$

- When $\Phi$, $s$ are sufficiently smooth, a quickly computable low-dimensional approximation of $s \circ \Phi$ should exist.
- **Idea 1:** State space reduction:
  - Define approximation $\Phi_N : \mathcal{P} \to V_N$ via Galerkin projection, $\dim V_N =: N \ll \dim V$.
  - Approximate $s \circ \Phi \approx s \circ \Phi_N$.
- **Idea 2:** $V_N \subseteq \operatorname{span}\{\Phi(\mu_1), \dots, \Phi(\mu_k)\}$.
- **Idea 3:** Construct $V_N$ iteratively via greedy search of $\mathcal{P}$ using quickly computable surrogate $\eta_N(\Phi_N(\mu), \mu) \geq \|\Phi(\mu) - \Phi_N(\mu)\|$.

# An Example

ulm university universität ulm

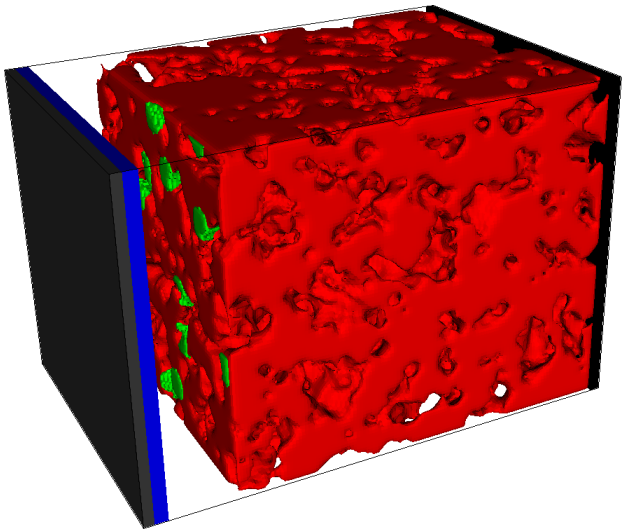
DLR Institute of Technical Thermodynamics

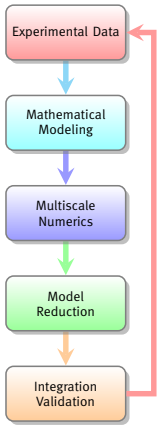
MULTIBAT

SPONSORED BY THE Federal Ministry of Education and Research

Fraunhofer ITWM

Westfälische Wilhelms-Universität Münster



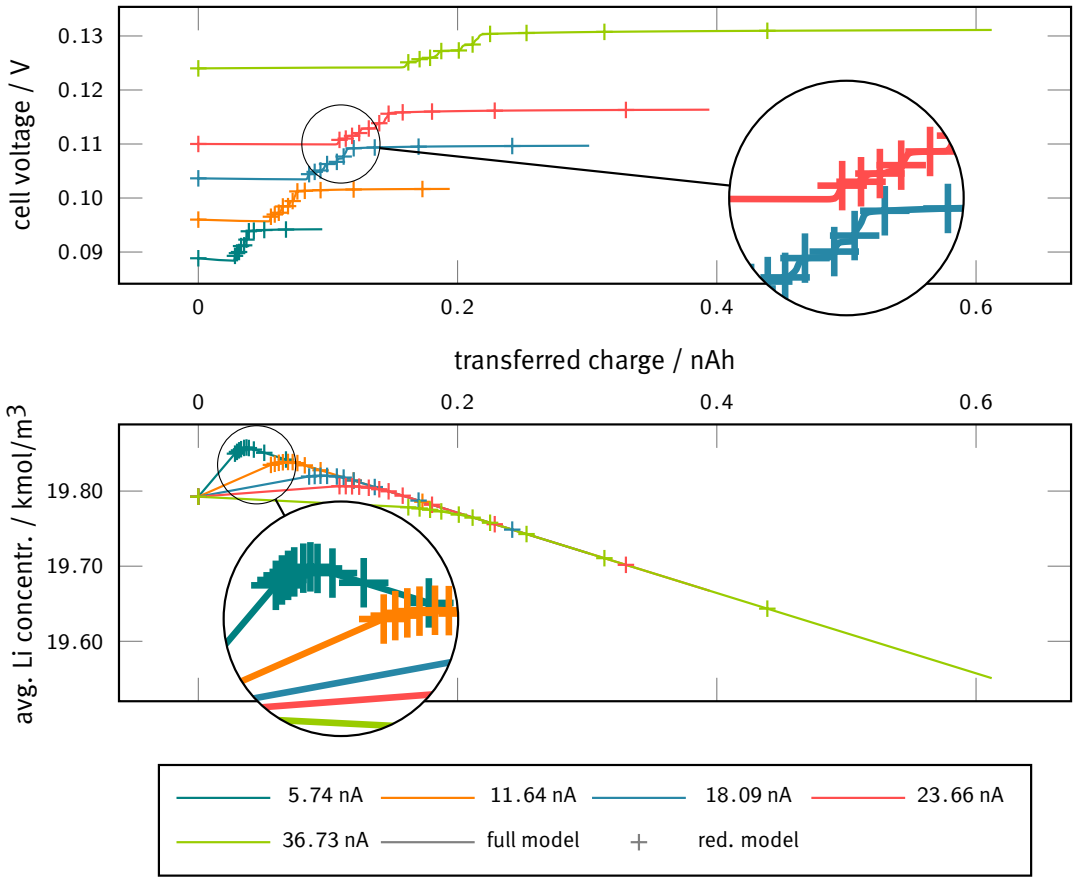



**MULTIBAT:** Gain understanding of degradation processes in rechargeable Li-Ion Batteries through mathematical modeling and simulation.

- Focus: Li-Plating.
- Li-plating initiated at interface between active particles and electrolyte.
- Need microscale models which resolve active particle geometry.
- Huge nonlinear discrete models.

# An Example – MOR Results

- 2.920.000 DOFs
- Snapshots: 3
- dim $V_N = 98 + 47$
- $M = 710 + 774$
- Rel. err.: $< 1.5 \cdot 10^{-3}$
- Full model: $\approx$ 13h
- Reduction: $\approx$ 9h
- Red. model: $\approx$ 5m
- Speedup: **154**



# Model Order Reduction with pyMOR

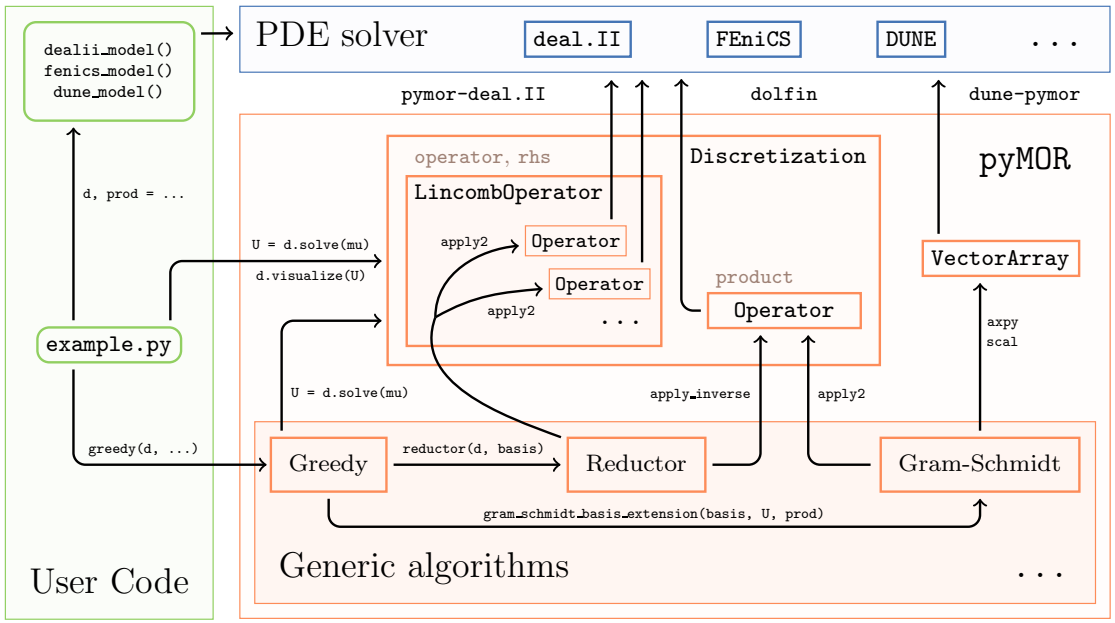
# pyMOR – Model Order Reduction with Python

## Goal

One tool for algorithm development *and* large-scale applications.

- Started late 2012, 20k lines of Python code, 3k single commits.
- BSD-licensed, fork us on Github!
- Quick prototyping with Python 2/3.
- Comes with small `NumPy`/`SciPy`-based discretization toolkit for getting started quickly.
- Seamless integration with high-performance PDE solvers.

# Generic Algorithms and Interfaces for MOR




- `VectorArray`, `Operator`, `Discretization` classes represent objects in solver's memory.
- No communication of high-dimensional data.
- Tight, low-level integration with external solver.
- No MOR-specific code in solver.

# Implemented Algorithms

- Gram-Schmidt, POD.
- Greedy basis generation with different extension algorithms.
- Automatic (Petrov-)Galerkin projection of arbitrarily nested affine combinations of operators.
- Interpolation of arbitrary (nonlinear) operators, EI-Greedy, DEIM.
- A posteriori error estimation.
- Iterative linear solvers, Newton algorithm, time-stepping algorithms.
- New! System theory Methods: balanced truncation, IRKA, ... (`sys-mor` branch)

# FEniCS Support Included

- Directly interfaces FEniCS
  LA backend, no copies needed.
- Use same MOR code with both backends!
- Only 150 SLOC for bindings.
- Thermal block demo:
  30 SLOC FEniCS +
  15 SLOC wrapping for pyMOR.
- Easily increase FEM order, etc.

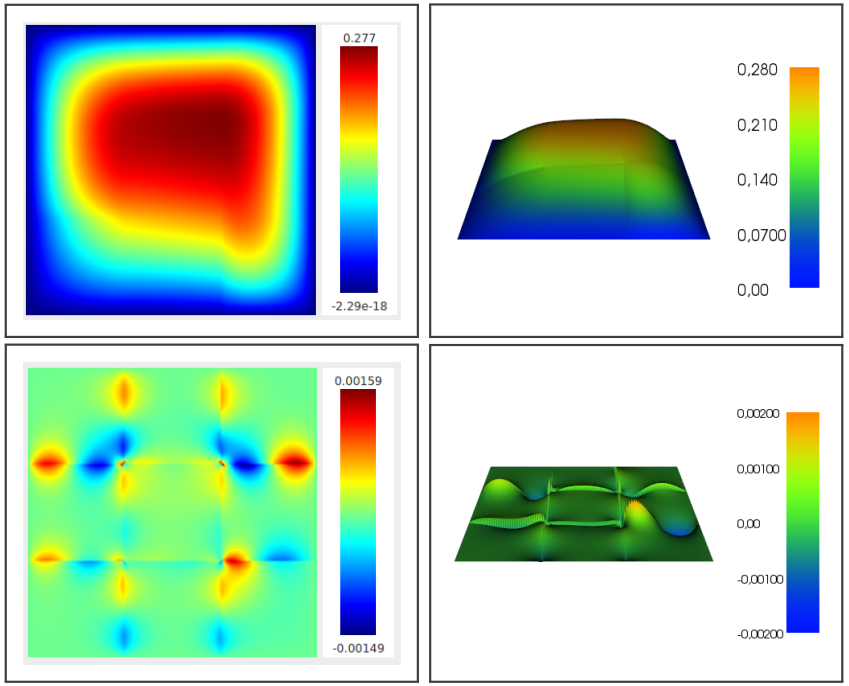



Figure: 3x3 thermal block problem
top: red. solution, bottom: red. error
left: pyMOR solver, right: FEniCS solver

# deal.II Support

- `pymor-deall.II` support module

  `https://github.com/pymor/pymor-deal.II`

- Python bindings for
  - `dealii::Vector`,
  - `dealii::SparseMatrix`.
- pyMOR wrapper classes.
- MOR demo for linear elasticity example from tutorial.

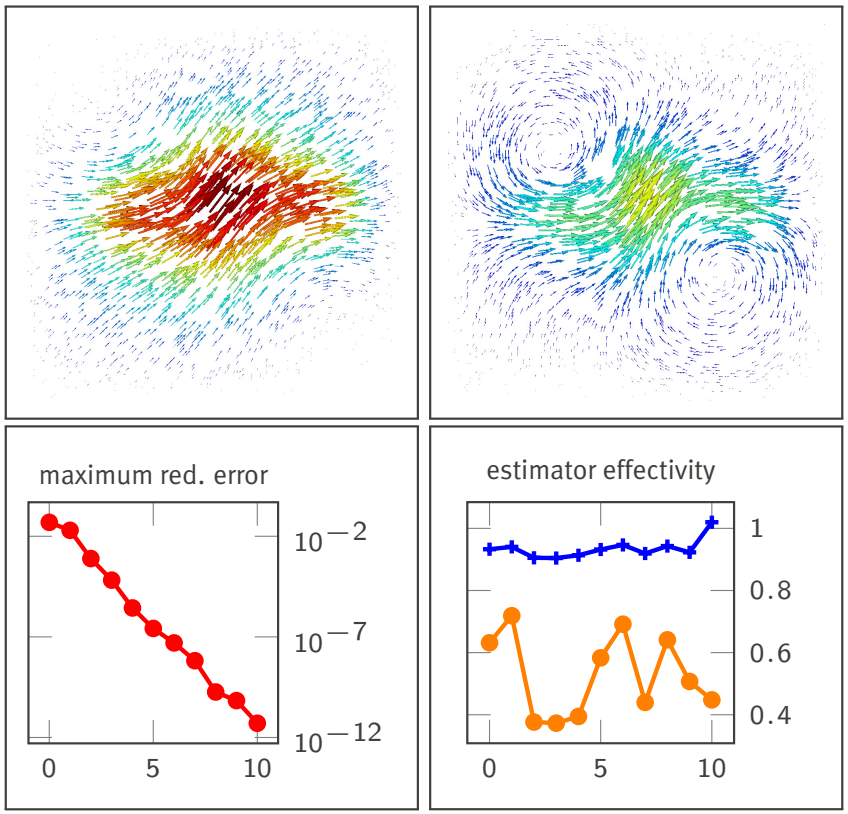



Figure: top: Solutions for $(\mu, \lambda) = (1, 1)$ and $(\mu, \lambda) = (1, 10)$, bottom: red. errs. and max./min. estimator effectivities vs. dim $V_N$.

# Tools for interfacing MPI parallel solvers

- Automatically make sequential bindings MPI aware.
- Reduce HPC-Cluster models without thinking about MPI at all.
- Interactively debug MPI parallel solvers.

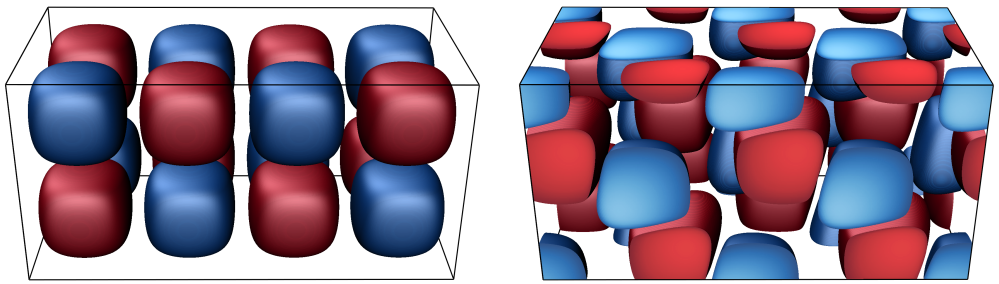


Figure: FV solution of 3D Burgers-type equation ($27.6 \cdot 10^6$ DOFs, 600 time steps) using Dune.

Table: Time (s) needed for solution using DUNE / DUNE with pyMOR timestepping.

| MPI ranks | 1 | 2 | 3 | 6 | 12 | 24 | 48 | 96 | 192 |
|---|---|---|---|---|---|---|---|---|---|
| DUNE | 17076 | 8519 | 5727 | 2969 | 1525 | 775 | 395 | 202 | 107 |
| pyMOR | 17742 | 8904 | 6014 | 3139 | 1606 | 816 | 418 | 213 | 120 |
| overhead | 3.9% | 4.5% | 5.0% | 5.7% | 5.3% | 5.3% | 6.0% | 5.4% | 11.8% |

# Model Order Reduction with pyMOR and NGSolve

# NGSolve Support

- Based on NGSolve - Python.
- See `ngs-user-meeting17` branch of pyMOR repo.
- pyMOR wrappers for `GridFunction` and `BaseMatrix` classes.
- 3d thermal block demo included.
- Joint work with Christoph Lehrenfeld.

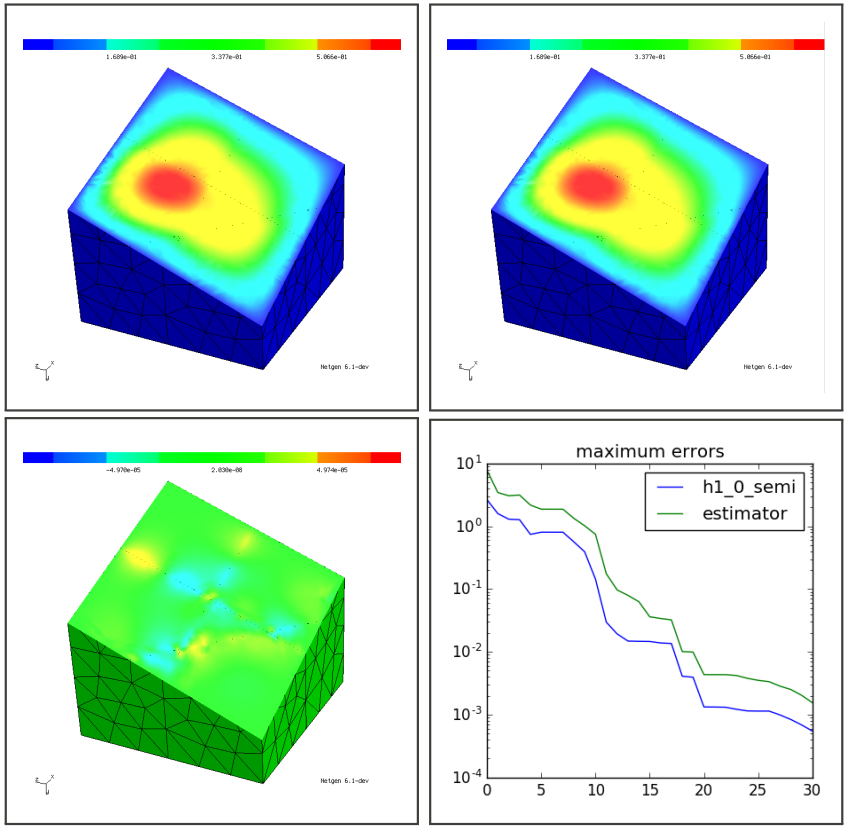



Figure: 3d thermal block problem
top: full/red. sol., bottom: err. for worst approx. $\mu$ and max. red. error vs. dim $V_N$.

```
import live_demo
```

# Matrix DEIM with NGSolve

## Osmotic cell swelling problem

$$\begin{aligned} \partial_t u - \alpha \Delta u &= 0 && \text{in } \Omega(t) \\ \mathcal{V}_n u + \alpha \partial_n u &= 0 && \text{on } \partial\Omega(t) \\ -\beta\kappa + \gamma(u - u_0) &= \mathcal{V}_n && \text{on } \partial\Omega(t) \end{aligned}$$

osmosis

## Transformed concentration equation on reference domain

$$\int_{\hat{\Omega}} J^{n+1} \hat{u}^{n+1} \hat{v} \, dx + \Delta t \int_{\hat{\Omega}} J^{n+1} V \cdot (F^{n+1^{-T}} \cdot \nabla_{\hat{x}} \hat{v}) \hat{u}^{n+1} dx$$
$$+ \Delta t \int_{\hat{\Omega}} \alpha J^{n+1} (F^{n+1^{-T}} \nabla_{\hat{x}} u) \cdot (F^{n+1^{-T}} \nabla_{\hat{x}} \hat{v}) \, dx = \int_{\hat{\Omega}} J^n \hat{u}^n \hat{v} \, dx$$

- Nonlinear in transformation $\Psi$, $F := \partial_x \Psi$, $J := |\det(F)|$, $V := \partial_t \Psi$.
- Use Matrix-DEIM w.r.t. $\Psi$.

# Thank you for your attention!

My homepage
`http://stephanrave.de/`

Jupyter notebook
`http://stephanrave.de/talks/pymor_ngsolve.ipynb`

pyMOR – Generic Algorithms and Interfaces for Model Order Reduction
SIAM J. Sci. Comput., 38(5), pp. S194–S216
`http://www.pymor.org/`

MULTIBAT: Unified Workflow for fast electrochemical 3D simulations of lithium-ion cells combining virtual stochastic microstructures, electrochemical degradation models and model order reduction (submitted)
`https://arxiv.org/abs/1704.04139`