Westfälische
Wilhelms-Universität
Münster

Model Order Reduction for the Masses

René Fritze, Petar Mlinarić, Stephan Rave, Felix Schindler

living.knowledge
WWU Münster

GAMM Jahrestagung 2019

Vienna

February 19, 2019

Outline

- What is Model Order Reduction?
- Model Order Reduction with pyMOR.
- Towards Model Order Reduction for the Masses.

pyMOR main developers

René Fritze

Petar Mlinarić

Stephan Rave

Felix Schindler

What is Model Order Reduction?

System-Theoretic Model Order Reduction

Linear time invariant system (full order model)

Input $u(t) \in \mathbb{R}^m$ to state $x(t) \in \mathbb{R}^n$ to output $y(t) \in \mathbb{R}^p$ mapping is given by

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t).\end{aligned}$$

System-Theoretic Model Order Reduction

Linear time invariant system (full order model)

Input $u(t) \in \mathbb{R}^m$ to state $x(t) \in \mathbb{R}^n$ to output $y(t) \in \mathbb{R}^p$ mapping is given by

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t).\end{aligned}$$

Model Reduction Magic

$$\text{Compute 'good' } W,\ V \in \mathbb{R}^{n\times r},\ r \ll n, \qquad s.t. \qquad W^T V = I$$

Linear time invariant system (reduced order model)

Input $u(t) \in \mathbb{R}^m$ to state $x(t) \in \mathbb{R}^r$ to output $y(t) \in \mathbb{R}^p$ mapping is given by

$$\begin{aligned}\dot{x}(t) &= (W^T AV)\,x(t) + (W^T B)\,u(t)\\ y(t) &= (CV)\,x(t).\end{aligned}$$

System-Theoretic MOR – Computing V and W

Balanced Truncation

- Compute reachability Gramian $\mathcal{P}$ and observability Gramian $\mathcal{Q}$ subject to

$$A\mathcal{P} + \mathcal{P}A^T + BB^T = 0$$
$$A^T\mathcal{Q} + \mathcal{Q}A + C^TC = 0.$$

- Simultaneously diagonalize $\mathcal{P}$ and $\mathcal{Q}$.
- Select V, W by truncating states which are both hard to reach and hard to observe.

System-Theoretic MOR – Computing V and W

Balanced Truncation

- Compute reachability Gramian $\mathcal{P}$ and observability Gramian $\mathcal{Q}$ subject to

$$A\mathcal{P} + \mathcal{P}A^T + BB^T = 0$$
$$A^T\mathcal{Q} + \mathcal{Q}A + C^TC = 0.$$

- Simultaneously diagonalize $\mathcal{P}$ and $\mathcal{Q}$.
- Select V, W by truncating states which are both hard to reach and hard to observe.

Rational interpolation

- Construct V, W, s.t. the transfer function (Laplace transform of impulse response)

$$H(s) = C(sI - A)^{-1}B$$

  is interpolated (including higher moments) at points $s_1, \dots, s_k$ by a rational function.
- Methods: Moment matching, Padé approximation, IRKA, ...
- Computed using rational Krylov methods.

Reduced Basis Methods

Parametric linear parabolic problem (full order model)

For given parameter $\mu \in \mathcal{P}$, find $u_\mu(t) \in V_h$ s.t.

$$\begin{aligned} u_\mu(0) &= u_0 \\ \langle v, \partial_t u_\mu(t)\rangle + b_\mu(v, u_\mu(t)) &= f(v) \qquad\qquad \forall v \in V_h, \\ y_\mu(t) &= g(u_\mu(t)). \end{aligned}$$

Reduced Basis Methods

Parametric linear parabolic problem (full order model)

For given parameter $\mu \in \mathcal{P}$, find $u_\mu(t) \in V_h$ s.t.

$$\begin{aligned} u_\mu(0) &= u_0 \\ \langle v, \partial_t u_\mu(t)\rangle + b_\mu(v, u_\mu(t)) &= f(v) \qquad\qquad \forall v \in V_h, \\ y_\mu(t) &= g(u_\mu(t)). \end{aligned}$$

Parametric linear parabolic problem (reduced order model)

For given $V_N \subset V_h$, let $u_{\mu,N}(t) \in V_N$ be given by Galerkin proj. onto V_N, i.e.

$$\begin{aligned} u_{\mu,N}(0) &= P_{V_N}(u_0), \\ \langle v, \partial_t u_{\mu(t),N}\rangle + b_\mu(v, u_{\mu(t),N}) &= f(v) \qquad\qquad \forall v \in V_N, \\ y_{\mu,N}(t) &= g(u_{\mu,N}(t)), \end{aligned}$$

where $P_{V_N} : V_h \to V_N$ is orthogonal proj. onto V_N.

RB Methods – Computing V_N

Weak greedy basis generation

```
1: function WEAK-GREEDY($\mathcal{S}_{train} \subset \mathcal{P}, \varepsilon$)
2:     $V_N \leftarrow \{0\}$
3:     while $\max_{\mu \in \mathcal{S}_{train}}$ ERR-EST(ROM-SOLVE($\mu$), $\mu$) $> \varepsilon$ do
4:         $\mu^* \leftarrow \text{arg-max}_{\mu \in \mathcal{S}_{train}}$ ERR-EST(ROM-SOLVE($\mu$), $\mu$)
5:         $V_N \leftarrow$ BASIS-EXT($V_N$, FOM-SOLVE($\mu^*$))
6:     end while
7:     return $V_N$
8: end function
```

BASIS-EXT

1. Compute $u_{\mu^*}^{\perp}(t) = (I - P_{V_N})u_{\mu^*}(t)$.
2. Add POD($u_{\mu^*}^{\perp}(t)$) to V_N (leading left-singular vectors of snapshot matrix).

ERR-EST

Use residual-based error estimate w.r.t. FOM (finite dimensional $\rightsquigarrow$ can compute dual norms).

RB Methods – Online Efficiency

Parametric linear parabolic problem (reduced order model)

$$\begin{aligned} u_{\mu,N}(0) &= P_{V_N}(u_0), \\ \langle v, \partial_t u_{\mu(t),N}\rangle + b_\mu(v, u_{\mu(t),N}) &= f(v) \qquad \forall v \in V_N, \\ y_{\mu,N}(t) &= g(u_{\mu,N}(t)), \end{aligned}$$

Affine decomposition

Assume that b_μ can be written as

$$b_\mu(v,u) = \sum_{q=1}^{Q} \theta_q(\mu) b_q(v,u).$$

RB Methods – Online Efficiency

Parametric linear parabolic problem (reduced order model)

$$\begin{aligned} u_{\mu,N}(0) &= P_{V_N}(u_0), \\ \langle v, \partial_t u_{\mu(t),N}\rangle + b_\mu(v, u_{\mu(t),N}) &= f(v) \qquad \forall v \in V_N, \\ y_{\mu,N}(t) &= g(u_{\mu,N}(t)), \end{aligned}$$

Affine decomposition

Assume that b_μ can be written as

$$b_\mu(v, u) = \sum_{q=1}^{Q} \theta_q(\mu) b_q(v, u).$$

Offline/Online splitting

By pre-computing

$$\langle \varphi_i, \varphi_j \rangle, \; b_q(\varphi_i, \varphi_j), \; f(\varphi_i), \; g(\varphi_i)$$

for a reduced basis $\varphi_1, \ldots, \varphi_N$ of V_N, solving ROM becomes independent of dim V_h.

RB Methods – Nonlinear Problems

Parametric nonlinear parabolic problem (full order model)

$$
\begin{aligned}
u_\mu(0) &= u_0, \\
\langle v, \partial_t u_\mu(t)\rangle + \langle v, \mathcal{A}_\mu(u_\mu(t))\rangle &= f(v) \qquad\qquad \forall v \in V_h, \\
y_\mu(t) &= g(u_\mu(t)),
\end{aligned}
$$

where $\mathcal{A}_\mu : V_h \to V_h^*$ is a nonlinear operator.

RB Methods – Nonlinear Problems

Parametric nonlinear parabolic problem (full order model)

$$\begin{aligned} u_\mu(0) &= u_0, \\ \langle v, \partial_t u_\mu(t)\rangle + \langle v, \mathcal{A}_\mu(u_\mu(t))\rangle &= f(v) \qquad\qquad \forall v \in V_h, \\ y_\mu(t) &= g(u_\mu(t)), \end{aligned}$$

where $\mathcal{A}_\mu : V_h \to V_h^*$ is a nonlinear operator.

Parametric nonlinear parabolic problem (reduced order model)

$$\begin{aligned} u_{\mu,N}(0) &= P_{V_N}(u_0), \\ \langle v, \partial_t u_{\mu(t),N}\rangle + \langle v, \mathcal{A}_\mu(u_{\mu(t),N})\rangle &= f(v) \qquad\qquad \forall v \in V_N, \\ y_{\mu,N}(t) &= g(u_{\mu,N}(t)). \end{aligned}$$

RB Methods – Nonlinear Problems

Parametric nonlinear parabolic problem (full order model)

$$\begin{aligned} u_\mu(0) &= u_0, \\ \langle v, \partial_t u_\mu(t)\rangle + \langle v, \mathcal{A}_\mu(u_\mu(t))\rangle &= f(v) \qquad \forall v \in V_h, \\ y_\mu(t) &= g(u_\mu(t)), \end{aligned}$$

where $\mathcal{A}_\mu : V_h \to V_h^*$ is a nonlinear operator.

Parametric nonlinear parabolic problem (reduced order model)

$$\begin{aligned} u_{\mu,N}(0) &= P_{V_N}(u_0), \\ \langle v, \partial_t u_{\mu(t),N}\rangle + \langle v, \mathcal{A}_\mu(u_{\mu(t),N})\rangle &= f(v) \qquad \forall v \in V_N, \\ y_{\mu,N}(t) &= g(u_{\mu,N}(t)). \end{aligned}$$

Problem: No offline/online splitting of nonlinear

$$\mathcal{A}_\mu : V_N \longrightarrow V_h \longrightarrow {V_N}^*.$$

Same problem for non-affinely decomposed b_μ.

RB Methods – Empirical Interpolation

EI – abstract version

Let normed Space V, functionals $\Psi \subseteq V^*$ and training set $\mathcal{M} \subset V$ be given.
Construct via EI-GREEDY algorithm:

1. Interpolation basis $b_1, \dots b_M \in \operatorname{span} \mathcal{M}$,
2. Interpolation functionals $\psi_1, \dots, \psi_M \in \Psi$.

The empirical interpolant $\mathcal{I}_M(v)$ of an arbitrary $v \in V$ is then determined by

$$\mathcal{I}_M(v) \in \operatorname{span}\{b_1, \dots, b_M\} \quad \text{and} \quad \psi_m(\mathcal{I}_M(v)) = \psi_m(v) \quad 1 \leq m \leq M.$$

EI Cheat Sheet

	V	Ψ	online
function EI	function space	point evaluations	evaluation at ‘magic points’
operator EI	range of (discrete) operator	DOFs	local evaluation at selected DOFs
matrix DEIM	matrices of given shape	matrix entries	assembly of selected entries

RB Methods – Hyper-Reduction

Reduced Order Model (with EI)

Find $u_{\mu,N} \in V_N$ s.t.

$$\langle v, \partial_t u_{\mu,N}(t)\rangle + \langle v, \{I_M \circ \mathcal{A}_{M,\mu} \circ R_{M'}\}(u_{\mu,N}(t))\rangle = f(v) \quad \forall v \in V_N,$$

where

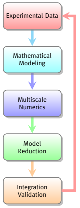
$R_{M'}\colon V_h \to \mathbb{R}^{M'}$	restriction to M' DOFs needed for local evaluation
$\mathcal{A}_{M,\mu}\colon \mathbb{R}^{M'} \to \mathbb{R}^M$	local evaluation of $\mathcal{A}_\mu$ at M interpolation DOFs
$I_M\colon \mathbb{R}^M \to V_h^*$	linear interpolation operator

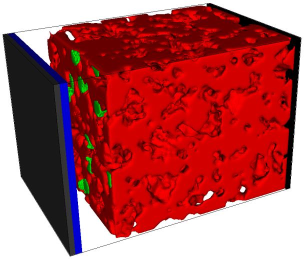
Offline/Online splitting

- Pre-compute the linear operators $\langle \cdot, I_M(\cdot)\rangle$ and $R_{M'}$ w.r.t. basis of V_N.
- Effort to evaluate $\langle \cdot, I_M \circ \mathcal{A}_{M,\mu} \circ R_{M'}(\cdot)\rangle$ w.r.t. this basis:

$$\mathcal{O}(MN) + \mathcal{O}(M) + \mathcal{O}(MN).$$

MULTIBAT: RB Approximation of Li-Ion Battery Models





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.
- Very large nonlinear discrete models.

MULTIBAT: Numerical Results

Model:

- Half-cell with plated Li
- μ = discharge current
- 2.920.000 DOFs

Reduction:

- Snapshots: 3
- $N = 178 + 67$
- $M = 924 + 997$
- Rel. err.: $< 4.5 \cdot 10^{-3}$

Timings:

- Full model: $\approx$ 15.5h
- Projection: $\approx$ 14h
- Red. model: $\approx$ 8m
- Speedup: **120**

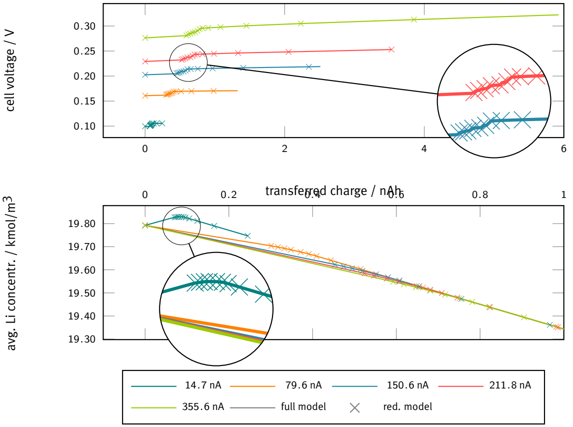


Figure: Validation of reduced order model output for random discharge currents; solid lines: full order model, markers: reduced order model.

Model Order Reduction with pyMOR

RB Software Designs

Design 1: write data needed for reduction to disk (e.g. `rbMIT`)

Handle reduction and reduced solving by dedicated MOR code. Read all needed data from disk after run of PDE solver in special MOR output mode.

Design 2: add MOR mode to PDE solver (e.g. `libMesh`)

Add all MOR code needed directly to the PDE solver. Optionally, implement specialized MOR version of solver to run on small devices.

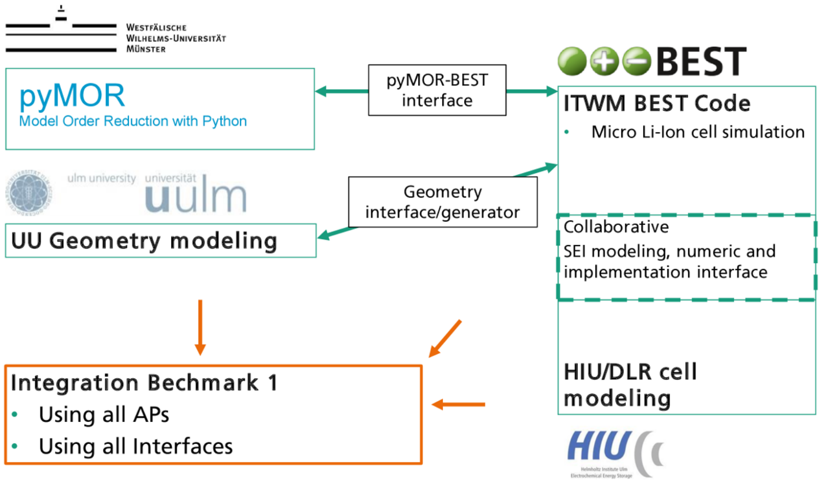
Design 3: communicate only reduced data (e.g. `RBmatlab + dune-rb`)

Write MOR code which communicates with running PDE solver. MOR code can, e.g., instruct solver to enrich basis with snapshot for certain μ and to compute data for reduced model.

RB Software Design Comparison

	via disk	in solver	low-dim
large (e.g. matrix-free) problems	🔴	🟢	🟢
empirical interpolation	🔴	🟢	🟡
reusability w. new solver	🟢	🔴	🟡
reusability w. new MOR alg.	🟢	🟢	🟡
MOR alg. easy to implement	🟢	🟡	🔴
easy to maintain	🟢	🟡	🔴
MOR and solver dev. decoupled	🟢	🔴	🔴

Software Interfaces in MULTIBAT


Westfälische Wilhelms-Universität Münster
pyMOR
Model Order Reduction with Python
pyMOR-BEST interface
BEST
ITWM BEST Code
Micro Li-Ion cell simulation
ulm university universität
uulm
Geometry interface/generator
UU Geometry modeling
Collaborative
SEI modeling, numeric and implementation interface
Integration Bechmark 1
Using all APs
Using all Interfaces
HIU/DLR cell modeling
HIU
Helmholtz Institute Ulm Electrochemical Energy Storage

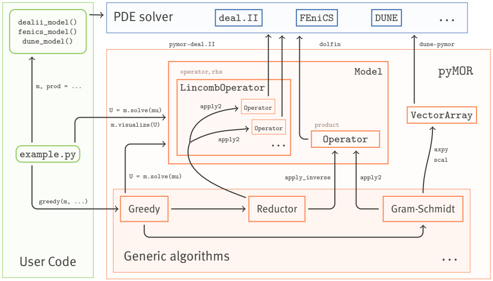
pyMOR – Model Order Reduction with Python

Goal

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

- Started late 2012, 24k lines of Python code, 4k single commits.
- BSD-licensed, fork us on GitHub!
- Quick prototyping with Python 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`, `Model` 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, HAPOD.
- 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, . . .

RB Software Design Comparison

	via disk	in solver	low-dim	pyMOR
large (e.g. matrix-free) problems	🔴	🟢	🟢	
empirical interpolation	🔴	🟢	🟡	
reusability w. new solver	🟢	🔴	🟡	
reusability w. new MOR alg.	🟢	🟢	🟡	
MOR alg. easy to implement	🟢	🟡	🔴	
easy to maintain	🟢	🟡	🔴	
MOR and solver dev. decoupled	🟢	🔴	🔴	

RB Software Design Comparison

	via disk	in solver	low-dim	pyMOR
large (e.g. matrix-free) problems	🔴	🟢	🟢	🟢
empirical interpolation	🔴	🟢	🟡	🟢
reusability w. new solver	🟢	🔴	🟡	🟢
reusability w. new MOR alg.	🟢	🟢	🟡	🟢
MOR alg. easy to implement	🟢	🟡	🔴	🟡
easy to maintain	🟢	🟡	🔴	🟢
MOR and solver dev. decoupled	🟢	🔴	🔴	🟢

FEniCS Support

- 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.

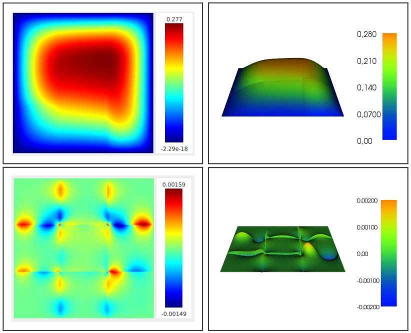


Figure: 3x3 thermal block problem
top: red. solution, bottom: red. error
left: pyMOR solver, right: FEniCS solver

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.

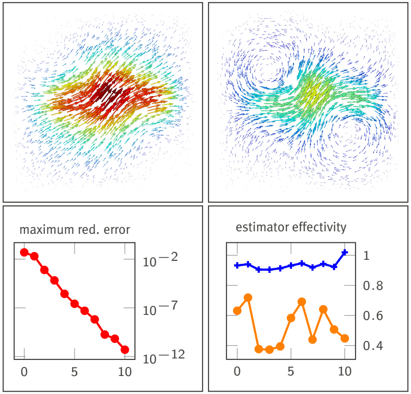


Figure: top: Solutions for $(\mu, \lambda) = (1, 1)$ and $(\mu, \lambda) = (1, 10)$, bottom: red. errs. and max./min. estimator effectivities vs. dim V_N.

NGSolve Support

- Based on `NGS-Py` Python bindings for NGSolve.
- pyMOR wrappers for vector and matrix classes.
- 3d thermal block demo included.
- Joint work with
  Christoph Lehrenfeld.

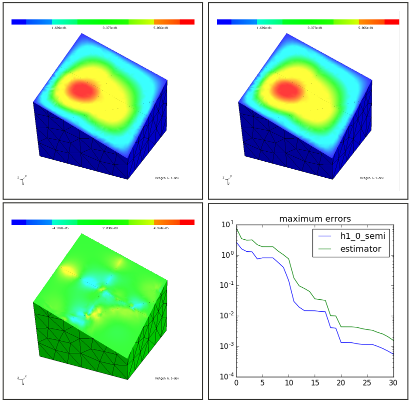
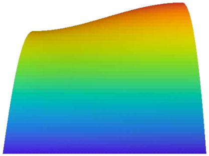


Figure: 3d thermal block problem
top: full/red. sol., bottom: err. for worst approx. μ and max. red. error vs. dim V_N.

Empirical Operator Interpolation with FEniCS



Two approaches for local operator evaluation:

1. Use `dolfin.assemble_local` (210 ms)
2. Use `dolfin.SubMesh` (35 ms)

See `fenics_nonlinear` branch.

Nonlinear Poisson problem from FEniCS docs (for $\mu = 1$)

$$\begin{aligned} -\nabla \cdot \{(1 + \mu u^2(x,y)) \cdot \nabla u(x,y)\} &= x \cdot \sin(y) && \text{for } x, y \in (0,1) \\ u(x,y) &= 1 && \text{for } x = 1 \\ \nabla u(x,y) \cdot n &= 0 && \text{otherwise} \end{aligned}$$

- `mesh = UnitSquareMesh(100, 100); V = FunctionSpace(mesh, "CG", 2)`
- Time for solution: 3.4 s
- $\mu \in [1, 1000]$, RB size: 3, EI DOFs: 5, rel. error $< 10^{-6}$

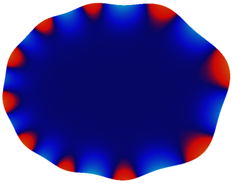
MOR for an NGSolve Free Boundary Problem

[Lehrenfeld, R, 19]

Osmotic cell swelling model [Lippoth, Prokert, 2012]

Given $\Omega(0) \subset \mathbb{R}^d$, $u(0) \in H^1(\Omega(0))$ and coefficients $u_{\text{ext}}, \alpha, \beta, \gamma \in \mathbb{R}$, the **concentration** $u(t)$ and **normal velocity** w_Γ of $\Gamma(t)$ is given by:

$$\begin{aligned} \partial_t u - \alpha \Delta u &= 0 && \text{in } \Omega(t), \\ w_\Gamma u + \alpha \partial_{\mathbf{n}} u &= 0 && \text{on } \Gamma(t), \\ -\beta\kappa + \gamma(u - u_{\text{ext}}) &= w_\Gamma && \text{on } \Gamma(t). \end{aligned}$$



- ALE formulation $\rightarrow$ diffusion coeffs nonlinear in deformation field Ψ
- Use EIM w.r.t. Ψ.

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.

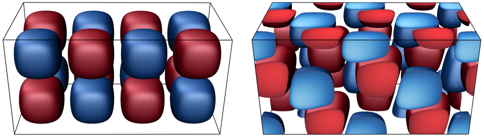


Figure: FV solution of 3D Burgers-type equation ($27.6 \cdot 10^6$ DOFs, 600 time steps) using Dune.

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%

Towards Model Order Reduction for the Masses

Challenges

MOR Researcher (Mathematician)

- ‘I’m a researcher not a software developer.’
- ‘I can quickly script all I need by myself.’
- ‘I don’t need complicated application problems.’

Challenges

MOR Researcher (Mathematician)

- ‘I’m a researcher not a software developer.’
- ‘I can quickly script all I need by myself.’
- ‘I don’t need complicated application problems.’

Potential User

- ‘I want things to go faster.’
- ‘`GenericPGReductor`!?!’
- ‘Don’t ask me which method to choose. You should now.’

Software Matters

- Software is a key tool for scientific discovery.
  - We need numerical experiments to explore the world.
  - We need numerical experiments to evaluate algorithms.
  - We also need to check large problems.

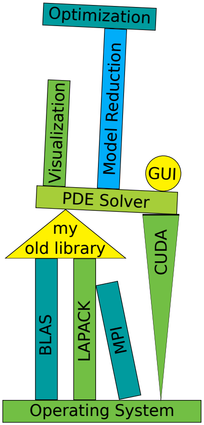
Software Matters

- Software is a key tool for scientific discovery.
  - We need numerical experiments to explore the world.
  - We need numerical experiments to evaluate algorithms.
  - We also need to check large problems.
- Software is a device for collaboration.
  - Basis for complex interdisciplinary simulation workflows.
  - What is an optimal method combining adaptive FEM, MOR and optimization?
  - Helps establish unified view on related methods.

Software Matters

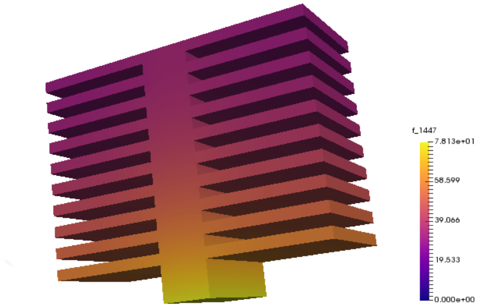
- Software is a key tool for scientific discovery.
  - We need numerical experiments to explore the world.
  - We need numerical experiments to evaluate algorithms.
  - We also need to check large problems.
- Software is a device for collaboration.
  - Basis for complex interdisciplinary simulation workflows.
  - What is an optimal method combining adaptive FEM, MOR and optimization?
  - Helps establish unified view on related methods.
- Software is costly.
  - Complexity has greatly increased.
  - Getting an algorithm ‘right’ takes effort.
  - Development time costs money/grad students.
  - Domain experts required.
  - We need more sustainable software development.

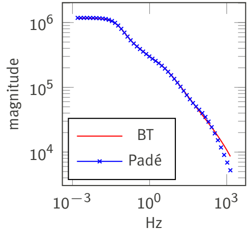
A Tower of Doom

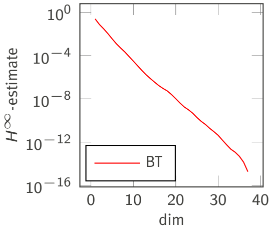


System-Theoretic MOR with FEniCS

- MPI distributed heatsink model with FEniCS
- Heat conduction with Robin boundary
- Input: heat flow at base
- Output: temperature at base
- MOR: Balanced truncation and Padé approximation







System-Theoretic MOR with FEniCS – Implementation

Model assembly with FEniCS

```
def discretize():
    domain = ...
    mesh = ms.generate_mesh(domain, RESOLUTION)
    subdomain_data = ...

    V = df.FunctionSpace(mesh, 'P', 1)
    u = df.TrialFunction(V)
    v = df.TestFunction(V)
    ds = df.Measure('ds', domain=mesh, subdomain_data=boundary_markers)

    A = df.assemble(- df.Constant(100.) * df.inner(df.grad(u), df.grad(v)) * df.dx
                    - df.Constant(0.1) * u * v * ds(1))
    B = df.assemble(df.Constant(1000.) * v * ds(2))
    E = df.assemble(u * v * df.dx)
```

System-Theoretic MOR with FEniCS – Implementation

pyMOR wrapping

```
# def discretize (cont.)
    # monkey patch apply_inverse_adjoint, assuming symmetriy
    FenicsMatrixOperator.apply_inverse_adjoint = FenicsMatrixOperator.apply_inverse

    space = FenicsVectorSpace(V)
    A = FenicsMatrixOperator(A, V, V)
    B = VectorOperator(space.make_array([B]))
    C = B.H
    E = FenicsMatrixOperator(E, V, V)
    fom = LTIModel(A, B, C, None, E)
```

System-Theoretic MOR with FEniCS – Implementation

pyMOR wrapping

```
# def discretize (cont.)
    # monkey patch apply_inverse_adjoint, assuming symmetriy
    FenicsMatrixOperator.apply_inverse_adjoint = FenicsMatrixOperator.apply_inverse

    space = FenicsVectorSpace(V)
    A = FenicsMatrixOperator(A, V, V)
    B = VectorOperator(space.make_array([B]))
    C = B.H
    E = FenicsMatrixOperator(E, V, V)
    fom = LTIModel(A, B, C, None, E)
```

MPI wrapping

```
from pymor.tools import mpi
if mpi.parallel:
    from pymor.models.mpi import mpi_wrap_model
    fom = mpi_wrap_model(discretize, use_with=True)
else:
    fom = discretize()
```

System-Theoretic MOR with FEniCS – Implementation

Balanced Truncation

```
reductor = BTReductor(fom)
bt_rom = reductor.reduce(10)

bt_rom.mag_plot(np.logspace(-2, 4, 100), Hz=True)
```

Padé approximation

```
r = 10
V = arnoldi(fom.A, fom.E, fom.B, [0] * r)
W = arnoldi(fom.A, fom.E, fom.C, [0] * r, trans=True)
pade_rom = GenericPGReductor(fom, W, V, False).reduce()

pade_rom.mag_plot(np.logspace(-2, 4, 100), Hz=True)
```

Bringing pyMOR to the Masses

a.k.a the greater Scientific Computing community

We need:

- more and better documentation and template examples.
- more extensive PDE Solver support.
- better frontend code.
- robust methods with few tunable parameters.
- more outreach to other communities.

Bringing pyMOR to the Masses

a.k.a the greater Scientific Computing community

We need:

- more and better documentation and template examples.
- more extensive PDE Solver support.
- better frontend code.
- robust methods with few tunable parameters.
- more outreach to other communities.

How to fund this?

- Does not produce research papers.

Bringing pyMOR to the Masses

a.k.a the greater Scientific Computing community

We need:

- more and better documentation and template examples.
- more extensive PDE Solver support.
- better frontend code.
- robust methods with few tunable parameters.
- more outreach to other communities.

How to fund this?

- Does not produce research papers.
- DFG call ‘Research Software Sustainability’.

Bringing pyMOR to the Masses

a.k.a the greater Scientific Computing community

We need:

- more and better documentation and template examples.
- more extensive PDE Solver support.
- better frontend code.
- robust methods with few tunable parameters.
- more outreach to other communities.

How to fund this?

- Does not produce research papers.
- DFG call ‘Research Software Sustainability’.
- pyMOR won!

pyMOR School

October 7-11, 2019
MPI Magdeburg
https://school.pymor.org

Thank you for your attention!

pyMOR – Generic Algorithms and Interfaces for Model Order Reduction
SIAM J. Sci. Comput., 38(5), 2016.
`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
J. Comp. Sci., 2018.

MS3: Research software and -data: How to ensure replicability, reproducibility, and reusability, today, 2:00pm - 4:00pm, HS 50

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