Model Order Reduction with Python

L. Balicki[1], R. Fritze[2], H. Kleikamp[2], P. Mlinarić[1], S. Rave[2], J. Saak[3], F. Schindler[2]

[1]Virginia Tech, Blacksburg, USA; [2]University of Münster, Germany; [3]Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany

Synopsis. pyMOR is a software library for building model order reduction applications with the Python programming language. Implemented algorithms include reduced basis methods for parametric linear and non-linear problems, as well as system-theoretic methods such as balanced truncation or IRKA and purely data-driven approches like DMD. All algorithms in pyMOR are formulated in terms of abstract interfaces for seamless integration with external PDE solver packages. Moreover, pure Python implementations of finite element and finite volume discretizations using the NumPy/SciPy scientific computing stack are provided for getting started quickly.

```
pip3   install pymor
conda  install -c conda-forge pymor
docker pull pymor/demo:main
open   https://bit.ly/3Zcf5jX
```

External solver support. pyMOR includes support for many popular PDE solver packages (NEW with experimental FEniCSx support). Integration with custom solvers is easily possible by exposing internal data structures and solution algorithms as `VectorArrays`, `Operators` and `Models`.

Join the community!

- try one of our interactive tutorials
- attend pyMOR school `https://school.pymor.org`
- ask for help on GitHub discussions
- fix a good first issue and win a free t-shirt[1]
- become a contributor or main developer

[1] Add *#CSE23* to your PR. First three merged PRs win.

Models

StationaryModel, InstationaryModel, LTIModel, PHLTIModel, SecondOrderModel, LinearDelayModel, BilinearModel, TransferFunction, QuadraticHamiltonianModel, LinearStochasticModel

Algorithms

POD, P-AAA (NEW), DEIM, DMD (NEW), IRKA, SAMDP, LGMRES, LSMR, LSQR, certified RB, HAPOD, TF-IRKA, rational Arnoldi, PSD cotangent lift (NEW), PSD complex SVD (NEW), modal truncation, time steppers, Slycot support, parametric PG projection, adaptive greedy basis generation, non-intrusive MOR with ANNs, low-rank ADI Lyapunov solver, low-rank ADI Riccati solver, bitangential Hermite interpolation, Gram-Schmidt with reiteration, symplectic Gram-Schmidt (NEW), PSD SVD-like decomposition (NEW), balanced truncation, empirical interpolation, Arnoldi eigensolver, randomized GSVD (NEW), randomized eigensolver (NEW), biorthogonal Gram-Schmidt, tangential rational Krylov, eigensys. realization alg. (NEW), second-order BT/IRKA

Model Operator VectorArray

Fork me on GitHub

Projecting an elliptic model.

Mathematical formulation

$$A(\mu)u(\mu) = F$$
$$A_r(\mu) := V^T A(\mu) V$$
$$F_r(\mu) := V^T F$$
$$A_r(\mu)u_r(\mu) = F_r$$

Low-level interface usage

```
u   = A_op.apply_inverse(F, mu)
A_r = V.inner(A_op.apply(V, mu))
F_r = V.inner(F)
u_r = np.linalg.solve(A_r, F_r)
```

Automatic projection (parameter-aware)

```
A_r_op = project(A, V, V)
```

High-level code

```
u   = fom.solve(mu)
red = StationaryRBReductor(fom, V)
rom = red.reduce()
u_r = rom.solve(mu)
```

Building a model.

Using NumPy/SciPy matrices:

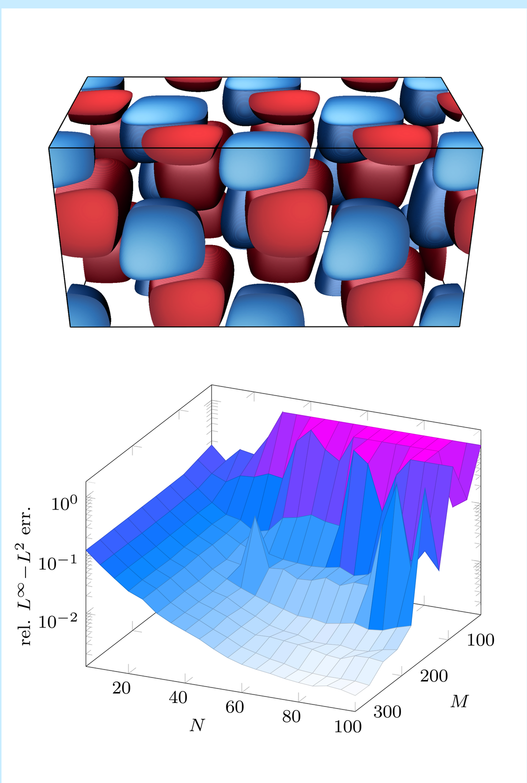
```
fom = LTIModel.from_matrices(A, B, C, D, E)
```

Using builtin discretization toolkit:

```
p = thermal_block_problem((2,3))  # or define your own problem
fom, data = discretize_stationary_cg(diameter=1/100)
```

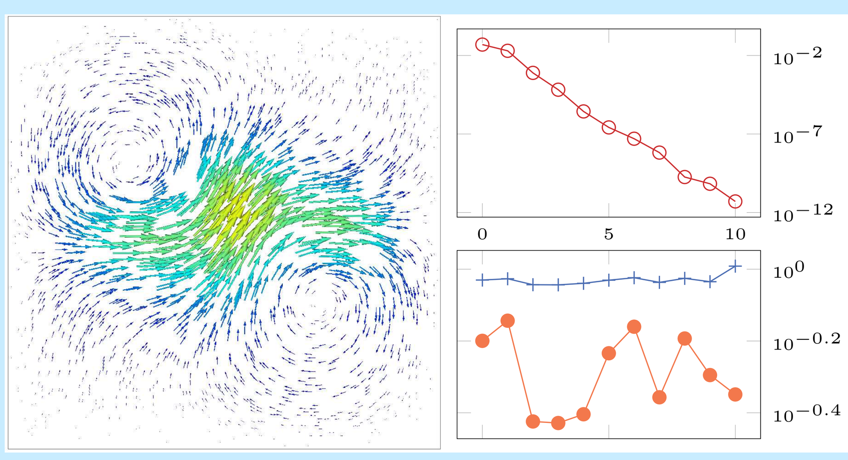
Using an external solver: NEW

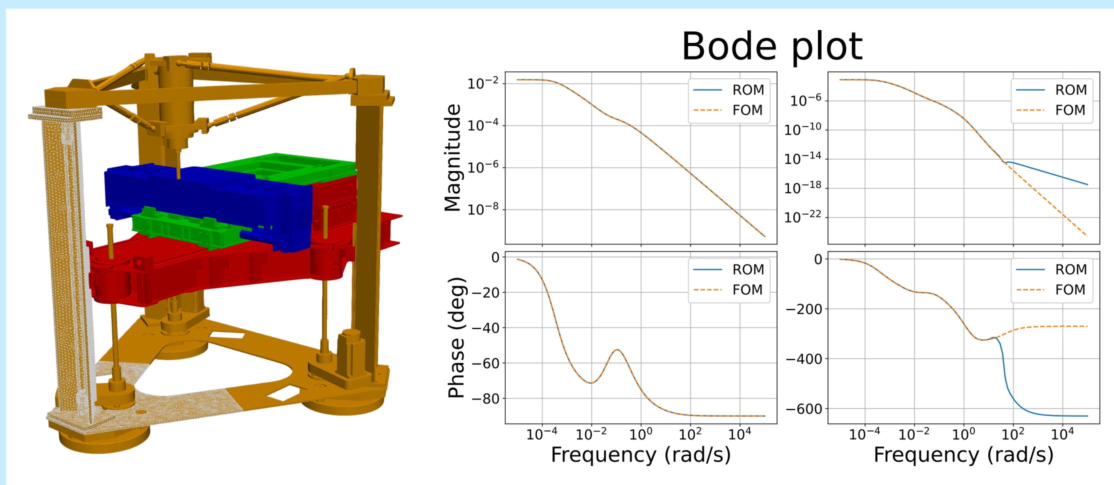
```
from pymor.discretizers.fenics import discretize_stationary_cg
fom, data = discretize_stationary_cg(p, degree=3)
```



◄ **MPI-distributed Burgers benchmark.** POD-DEIM reduction of a 3d Burgers-type benchmark problem implemented in DUNE; FV discretization with 27.6M DOFs distributed over 192 MPI ranks; top: simulation at final time; bottom: MOR errors vs. POD and DEIM basis sizes.

▼ **deal.II Step-08 Linear Elasticity Problem.** RB approximation of a linear elasticity problem parameterized by Lamé constants λ, μ using an adaptive greedy sampling strategy; left: displacement field for $(\mu, \lambda) = (1, 10)$; top right: maximum MOR error vs. basis size; bottom right: min/max error estimator efficiency vs. basis size.





▲ **Thermal Transregio 96 Model.** Thermal model which describes heat induced by friction and surrounding building blocks in one of the 'z-pillars'. The full-order model coming from a FEM discretization is represented by a linear state-space system with 39,527 DOFs. A ROM with order 30 has been computed using balanced truncation.