MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

`morgen` (1.1)

Model Order Reduction for Gas and Energy Networks

C. Himpe, S. Grundel

CSC

`https://git.io/morgen`

The Name `morgen`

`morgen` (german for: tomorrow)

- **M**odel
- **O**rder
- **R**eduction

for **G**as

and **E**nergy

- **N**etworks

Why `morgen`

- Testing model-solver-reductor combinations
- Comparing models, solvers, or reductors
- Benchmarking reductors
- Prototyping algorithms
- Uncertainty quantification

About `morgen`

- Open-source (BSD-2-Clause)
- High-Level (MATLAB and OCTAVE)
- Modular (Six modules)
- Configurable (Three-level configuration)
- Extensible (Contributions welcome)

morgen Modules

1. Models (Discretizations)
2. Solvers (Integrators)
3. Reductors (Model Reduction Algorithms)
4. Networks (Topologies & Scenarios)
5. Tests (Simulation and Model Reduction Experiments)
6. Tools (Unit and Format Converters)

Models

Implemented:

- `ode_mid` – midpoint discretization
- `ode_end` – endpoint discretization (port-Hamiltonian)

```
discrete = model(network,config);
```

`discrete` **Structure:**

- `.E` – Mass matrix function
- `.A` – System matrix
- `.B` – Input matrix
- `.C` – Output matrix
- `.f` – Nonlinear vector field
- `.J` – Vector field Jacobian
- `.x0` – Initial state
- `.nP` – Number of pressure states
- `.nQ` – Number of mass-flux states
- `.nPorts` – Number of ports

Solvers

Implemented:

- `imex1` – 1st Order Implicit-Explicit (Euler-Euler)
- `imex2` – 2nd Order Implicit-Explicit (Runge-Kutta)
- `generic` – 2nd Order Adaptive Rosenbrock (`ode23s`)
- `rk4` – “Classic” 4th Order Explicit Runge-Kutta
- `rk2hyp` – 2nd Order Explicit Runge-Kutta with increased stability
- `rk4hyp` – 4th Order Explicit Runge-Kutta with increased stability

```
solution = solver(discrete,scenario,config);
```

`solution` **Structure:**

- `.t` – Time instances
- `.u` – Input time series
- `.y` – Output time series
- `.runtime` – Solver runtime
- `steady_z0` – Mean compressibility
- `steady_error` – Steady-state error
- `steady_iter1` – Algebraic Iterations
- `steady_iter2` – Differential Iterations

Reductors

Implemented:

- `pod_r` – Structured Proper Orthogonal Decomposition
- `eds_ro` , `eds_wx` , `eds_wz` , `eds_ro_l` , `eds_wx_l` , `eds_wz_l` – Structured Dominant Subspaces Variants
- `bpod_ro` , `bpod_ro_l` – Structured Balanced Proper Orthogonal Decomposition
- `ebt_ro` , `ebt_wx` , `ebt_wz` , `ebt_ro_l` , `ebt_wx_l` , `ebt_wz_l` – Structured Balanced Truncation Variants
- `gopod_r` – Goal-Oriented Proper Orthogonal Decomposition
- `ebg_ro` , `ebg_wx` , `ebg_wz` , `ebg_ro_l` , `ebg_wx_l` , `ebg_wz_l` – Structured Balanced Gains Variants
- `dmd_r` – Dynamic Mode Decomposition Galerkin

```
[proj,name] = reductor(solver,discrete,scenario,config);
```

- `proj` – Cell array of projectors
- `name` – Full name of reductor

Networks & Tests

Network and scenario data:

- Network data stored as decorated edge list in CSV format (`.net`).
- Scenario data stored as key-vale pairs in INI format (`.ini`).
- Network base name determines associated scenario folder name.
- Each network has minimally a `training.ini` scenario.

Types of tests:

- Prefix `sim_` – Simulate scenario by a model-solver combination.
- Prefix `mor_` – Reduce and test model-solver-reductor combination.

Tools

Available:

- `xml2net` – Convert *GasLib* `.xml` to `morgen` `.net`
- `json2net` – Convert *MathEnergy* `.json` to `morgen` `.net`
- `csv2net` – Convert *SciGRID_gas* `.csv` to `morgen` `.net`
- `vf2kgs` – Convert volume flow to mass flow in kg/s
- `psi2bar` – Convert pressure from psi to bar
- `cmp_friction` – Compare friction factors
- `cmp_compressibility` – Compare compressibility factors
- `randscen` – Generate random scenario from training scenario

Configuration

Available:

- optional arguments (`varargin`)
- configuration file (`morgen.ini`)
- fallback via hard-coded default values

Use morgen

```
R = morgen(network_id,scenario_id,model_id,solver_id,reductor_ids,varargin);
```

{string} network_id – Network file (.net) base name

{string} scenario_id – Scenario file (.ini) base name

{string} model_id – Model function name

{string} solver_id – Solver function name

{cell} reductor_ids – Array of reductor names

{string} varargin – Adhoc configuration arguments (‘key=val‘)

Notes

- `morgen` is open source (under BSD-2-Clause license),
- and compatible with MATLAB and Octave.
- A template model, solver and reductor are available.
- Currently, all reductors use `emgr`: `https://gramian.de`.
- See `README.md` for more info.
- This is research software!

Summary

`morgen` – **M**odel **O**rder **R**eduction for **G**as and **E**nergy **N**etworks

$\rightarrow$ https://git.io/morgen $\leftarrow$

C. Himpe, S. Grundel, P. Benner:
Model Order Reduction for Gas and Energy Networks.
Journal of Mathematics in Industry 11: 13, 2021.
https://doi.org/10.1186/s13362-021-00109-4