Model Reduction for Advection dominated (Hyperbolic) problems

Davide Torlo

Universität Zürich
Institut für Mathematik

15th May 2020, “Trieste”

Outline

Outline

Motivation: parametric hyperbolic systems

$$\begin{cases} \partial_t u(x,t,\boldsymbol{\mu}) + \nabla \cdot F(u,x,t,\boldsymbol{\mu}) = 0, \quad x \in D,\, t \in \mathbb{R}^+,\, \boldsymbol{\mu} \in \mathcal{P} \subset \mathbb{R}^P \\ \mathbf{B}(u,\boldsymbol{\mu}) = g(t,\boldsymbol{\mu}) \\ u(x,t=0,\boldsymbol{\mu}) = u_0(x,\boldsymbol{\mu}) \end{cases} \tag{1}$$

- F non linear dependence on $\boldsymbol{\mu}$!
- $\boldsymbol{\mu}$ can influence boundaries, flux, initial conditions

- Why: many physical applications (fluid equations, kinetic models, etc.)
- Classical solvers: FV, FEM, FD, RD. (Slow for high–resolution)
- Many query task (UQ, optimization, etc.)

MOR: Ingredients

- Discretized solution $u_{\mathcal{N}}(\cdot, t, \boldsymbol{\mu}) \in \mathbb{V}_{\mathcal{N}}$ for $t \in \mathbb{R}^+$, $\boldsymbol{\mu} \in \mathcal{P}$
- Solution manifold: $\mathcal{S} := \{u_{\mathcal{N}}(\cdot, t, \boldsymbol{\mu}) \in \mathbb{V}_{\mathcal{N}} : t \in \mathbb{R}^+, \boldsymbol{\mu} \in \mathcal{P}\}$
- Ansatz:

$$\mathcal{S} \approx \mathbb{V}_{N_{RB}} \subset \mathbb{V}_{\mathcal{N}}, \qquad N_{RB} \ll \mathcal{N} \tag{2}$$

- Example: diffusion equation $u_t + \mu u_{xx} = 0$ with $u_0 = \sin(x\pi)$

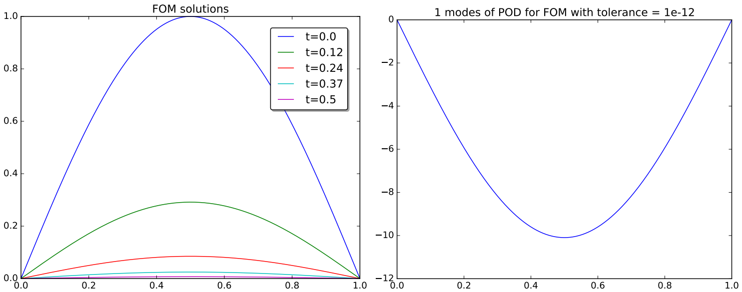


Figure: POD on a diffusion problem

MOR: Ingredients

Problem:

$$U^{n+1}(\boldsymbol{\mu}) - U^{n}(\boldsymbol{\mu}) + \mathcal{L}^{n}(U^{n}, \boldsymbol{\mu}) = 0, \quad U^{n}, U^{n+1} \in \mathbb{V}_{\mathcal{N}} \tag{3}$$

Objective:

$$\sum_{i=1}^{N_{RB}} \mathrm{u}_i^{n+1}(\boldsymbol{\mu})\psi_{RB}^{i} - \mathrm{u}_i^{n}(\boldsymbol{\mu})\psi_{RB}^{i} + \sum_{i=1}^{N_{RB}} \mathrm{L}^{i}(\mathrm{u}^{n}, \boldsymbol{\mu})\psi_{RB}^{i} = 0, \qquad \psi_{RB}^{i} \in \mathbb{V}_{\mathcal{N}}, \mathrm{u}^{n}, \mathrm{u}^{n+1} \in \mathbb{V}_{N_{RB}} \tag{4}$$

- EIM $\Rightarrow$ non–linear fluxes and scheme $\mathrm{L}^{i}(\mathrm{u}^{n}, \boldsymbol{\mu})$
- POD $\Rightarrow$ create the RB space and span the time evolution
- Greedy $\Rightarrow$ span the parameter space

Offline Algorithm: PODEIM–Greedy [1]

INITIALIZATION:

- EIM on $\mathcal{L}(U^n, \boldsymbol{\mu}_0, t^n)$ for $n \leq N_t$
- $RB = POD(\{U^n(\boldsymbol{\mu}_0)\}_{n=0}^{N_t})$

ITERATION:

- Greedy algorithm spanning over the parameter space $\mathcal{P}_h$, with an error indicator $\varepsilon(\mathbf{U}(\boldsymbol{\mu}))$ where $\mathbf{U}(\boldsymbol{\mu}) \in \mathbb{R}^N \times \mathbb{R}^+$
- Choose worst parameter as $\boldsymbol{\mu}^* = \arg\max_{\boldsymbol{\mu} \in \mathcal{P}_h} \varepsilon(\mathbf{U}(\boldsymbol{\mu}))$
- Apply POD on time evolution of selected solution $POD_{add} = POD\left(\{U^n(\boldsymbol{\mu}^*)\}_{n=1}^{N_t}\right)$
- Update the RB with $RB = POD\left(RB \cup POD_{add}\right)$
- Update EIM basis function with $EIM_{space} = EIM_{space} \cup EIM(\{\mathcal{L}(U^n, \boldsymbol{\mu}^*, t^n)\}_{n=0}^{N_t})$

[1] B. Haasdonk and M. Ohlberger, in Hyperbolic problems: theory, numerics and applications, vol. 67, Amer. Math. Soc., 2009.

Online algorithm: PODEIM–Greedy

Solve the smaller system:

$$\sum_{i=1}^{N_{RB}} (\mathrm{u}_i^{n+1}(\boldsymbol{\mu}) - \mathrm{u}_i^n(\boldsymbol{\mu}))\psi_{RB}^i + \sum_{i=1}^{N_{RB}} \sum_{j=1}^{N_{EIM}} \tau_j(\mathcal{L}(U^n, \boldsymbol{\mu}))\Pi_{RB,i}(f_j)\psi_{RB}^i = 0$$

- $\Pi_{RB,i}(f_j)$ are the projection on RB of the EIM functions: offline
- $\tau_j(\mathcal{L}(U^n, \boldsymbol{\mu}))$ are inexpensive to compute, but depend on the method (for RD $\approx \mathcal{O}(d)$)
- MOR cost $\mathcal{O}(N_t N_{RB} N_{EIM})$ vs FOM cost $\mathcal{O}(N_t \mathcal{N})$
- Gain if $N_{RB}, N_{EIM} \ll \mathcal{N}$
- Error estimator

Outline

Travelling wave, time evolution solution

$$\partial_t u + \partial_x u = 0$$

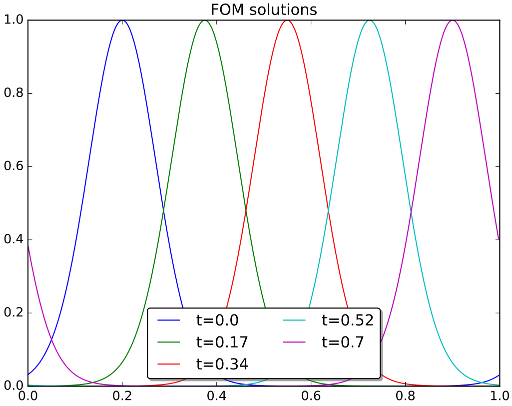


Figure: Solution of advection equation with wave IC

Travelling wave, POD

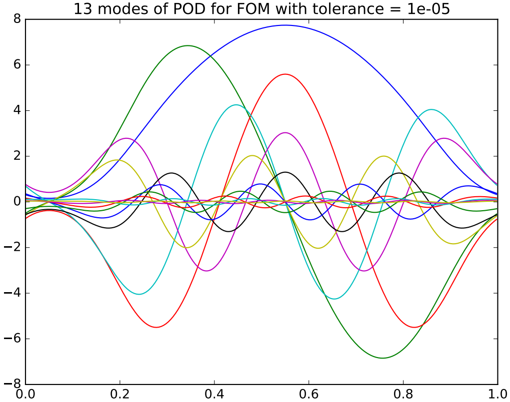


Figure: Solution of advection equation with wave IC

Travelling shock, time evolution solution, little diffusion

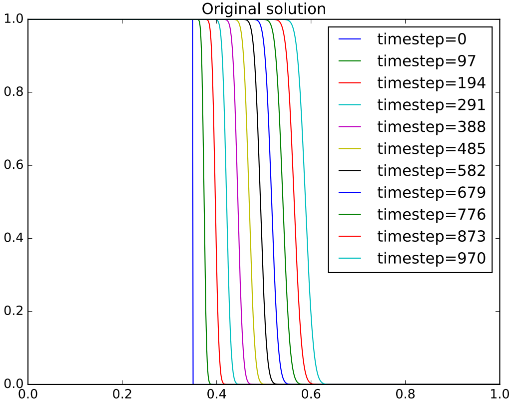


Figure: Solution of advection equation with shock IC

Travelling shock, POD, little diffusion

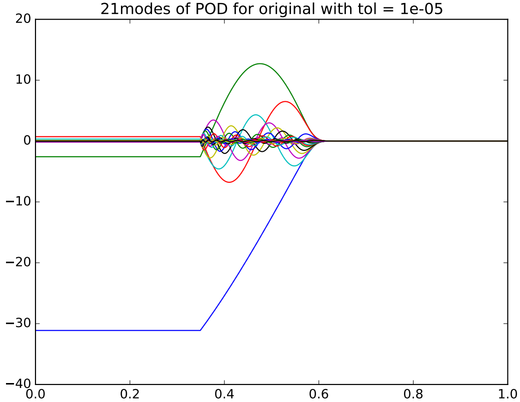


Figure: POD of time evolution of advection equation with shock IC

Travelling shock, time evolution solution, no diffusion

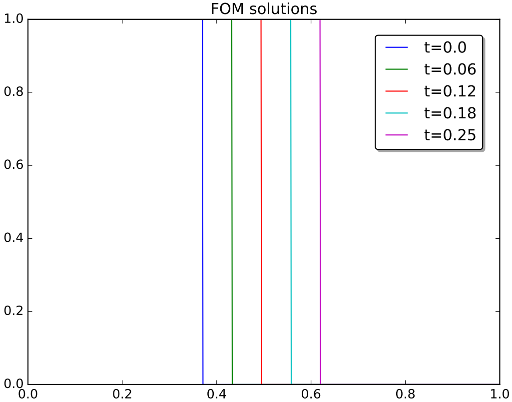


Figure: Solution of advection equation with shock IC

Travelling shock, POD, no diffusion

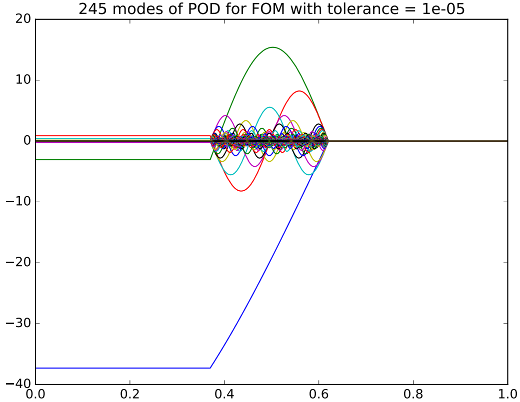


Figure: POD of time evolution of advection equation with shock IC

Common problems and properties

- As many basis functions as positions of the shock
- Slow decay of Kolmogorov N–width

$$d_N(\mathcal{S}, \mathbb{V}) := \inf_{\mathbb{V}_N \subset \mathbb{V}} \sup_{f \in \mathcal{S}} \inf_{g \in \mathbb{V}_N} ||f - g||$$

- Non linear dependency leads to big EIM and RB space
- 1/2 parameters problem (highly non linear dependence on parameters)

Outline

Solutions or partial solutions

Some possibilities to incorporate the advection into RB framework

- Freezing **Ohlberger, M. and Rave, S.**
- Shifted POD **Reiss, J., Schulze, P., Sesterhenn, J. and Mehrmann, V.**
- Lagrangian basis method **Mojgani, R. and Balajewicz, M.**
- Advection modes by optimal mass transport **Iollo, A. and Lombardi, D.**
- Calibration (also 2D non–periodic boundaries) **Cagniart, N., Stamm, B. and Maday, Y.**, **Crisovan, R. and Abgrall, R.**
- Online adaptive bases and samplings **Peherstorfer, B.**
- Transport Reversal **Rim, D., Moe, S. and LeVeque R. J.**
- Registration method **Taddei, T.**
- Preprocessing reduced basis **Karatzas,E., Nonino, M., Ballarin, F., Rozza, G. and Maday, Y.**
- Manifold learning via Neural Network **Carlberg, K. and Lee, K.; Lye, K., Mishra S. and Ray, D.; Fresca, S., Dedè, L. and Manzoni, A.**
- Dynamic Modes **Lu, H. and Tartakovsky, D. M.**

Solutions or partial solutions

Some possibilities to incorporate the advection into RB framework

- Freezing **Ohlberger, M. and Rave, S.**
- Shifted POD **Reiss, J., Schulze, P., Sesterhenn, J. and Mehrmann, V.**
- Lagrangian basis method **Mojgani, R. and Balajewicz, M.**
- Advection modes by optimal mass transport **Iollo, A. and Lombardi, D.**
- Calibration (also 2D non–periodic boundaries) **Cagniart, N., Stamm, B. and Maday, Y.**, **Crisovan, R. and Abgrall, R.**
- Online adaptive bases and samplings **Peherstorfer, B.**
- Transport Reversal **Rim, D., Moe, S. and LeVeque R. J.**
- Registration method **Taddei, T.**
- Preprocessing reduced basis **Karatzas,E., Nonino, M., Ballarin, F., Rozza, G. and Maday, Y.**
- Manifold learning via Neural Network **Carlberg, K. and Lee, K.; Lye, K., Mishra S. and Ray, D.; Fresca, S., Dedè, L. and Manzoni, A.**
- Dynamic Modes **Lu, H. and Tartakovsky, D. M.**

Solutions or partial solutions

Some possibilities to incorporate the advection into RB framework

- Freezing **Ohlberger, M. and Rave, S.**
- Shifted POD **Reiss, J., Schulze, P., Sesterhenn, J. and Mehrmann, V.**
- Lagrangian basis method **Mojgani, R. and Balajewicz, M.**
- Advection modes by optimal mass transport **Iollo, A. and Lombardi, D.**
- Calibration (also 2D non–periodic boundaries) **Cagniart, N., Stamm, B. and Maday, Y.**, **Crisovan, R. and Abgrall, R.**
- Online adaptive bases and samplings **Peherstorfer, B.**
- Transport Reversal **Rim, D., Moe, S. and LeVeque R. J.**
- Registration method **Taddei, T.**
- Preprocessing reduced basis **Karatzas,E., Nonino, M., Ballarin, F., Rozza, G. and Maday, Y.**
- Manifold learning via Neural Network **Carlberg, K. and Lee, K.; Lye, K., Mishra S. and Ray, D.; Fresca, S., Dedè, L. and Manzoni, A.**
- Dynamic Modes **Lu, H. and Tartakovsky, D. M.**

Solutions or partial solutions

Some possibilities to incorporate the advection into RB framework

- Freezing **Ohlberger, M. and Rave, S.**
- Shifted POD **Reiss, J., Schulze, P., Sesterhenn, J. and Mehrmann, V.**
- Lagrangian basis method **Mojgani, R. and Balajewicz, M.**
- Advection modes by optimal mass transport **Iollo, A. and Lombardi, D.**
- Calibration (also 2D non–periodic boundaries) **Cagniart, N., Stamm, B. and Maday, Y.**, **Crisovan, R. and Abgrall, R.**
- Online adaptive bases and samplings **Peherstorfer, B.**
- Transport Reversal **Rim, D., Moe, S. and LeVeque R. J.**
- Registration method **Taddei, T.**
- Preprocessing reduced basis **Karatzas,E., Nonino, M., Ballarin, F., Rozza, G. and Maday, Y.**
- Manifold learning via Neural Network **Carlberg, K. and Lee, K.; Lye, K., Mishra S. and Ray, D.; Fresca, S., Dedè, L. and Manzoni, A.**
- Dynamic Modes **Lu, H. and Tartakovsky, D. M.**

Transfomation of the domain

Let's suppose that there exists a *geometry* map

$$T : \Theta \times \mathcal{R} \to \Omega \tag{5}$$

- $T(\cdot,\cdot) \in \mathcal{C}^1(\Theta \times \mathcal{R}, \Omega)$,
- $\exists\, T^{-1} : \Theta \times \Omega \to \mathcal{R}$ such that $T^{-1}(\theta, T(\theta, y)) = y$ for $y \in \mathcal{R}$ and $T(\theta, T^{-1}(\theta, x)) = x$ for $x \in \Omega$,
- $T^{-1}(\cdot,\cdot) \in \mathcal{C}^1(\Theta \times \Omega, \mathcal{R})$.

Moreover, suppose that there exists a *calibration* map

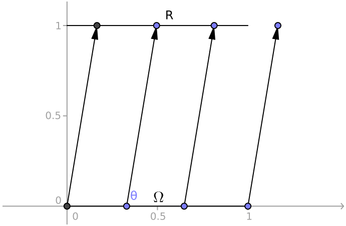
$$\theta : \mathcal{P} \times [0, t_f] \to \Theta$$

- $\theta(\cdot, \boldsymbol{\mu}) \in \mathcal{C}^1([0, t_f], \Theta)$ for all $\boldsymbol{\mu} \in \mathcal{P}$,
- $u_{\mathcal{N}}(T(\theta(t, \boldsymbol{\mu}), y), t, \boldsymbol{\mu}) \approx \bar{v}(y), \quad \forall \boldsymbol{\mu} \in \mathcal{P},\ t \in [0, t_f], y \in \mathcal{R}$

Transformation map for MOR

Examples: θ is the point of maximum height or of steepest solution.

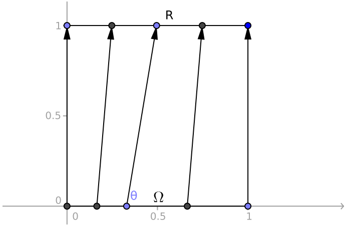
- Translation:
  $T(\theta, y) = y + \theta - 0.5$
  $T^{-1}(\theta, x) = x - \theta + 0.5$
- Dilatation:
  $T(\theta, y) = \frac{y\theta}{(2\theta-1)y+1-\theta}$
  $T^{-1}(\theta, x) = \frac{x(\theta-1)}{(2\theta-1)x-\theta}$
- Higher degree polynomials
- Gordon-Hall



Transformation map for MOR

Examples: θ is the point of maximum height or of steepest solution.

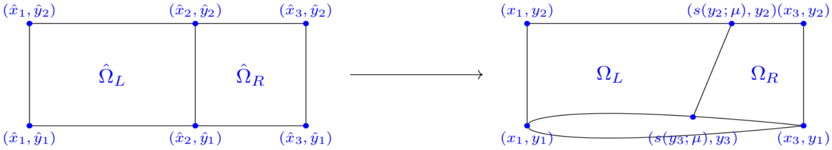
- Translation:
  $T(\theta, y) = y + \theta - 0.5$
  $T^{-1}(\theta, x) = x - \theta + 0.5$
- Dilatation:
  $T(\theta, y) = \frac{y\theta}{(2\theta - 1)y + 1 - \theta}$
  $T^{-1}(\theta, x) = \frac{x(\theta-1)}{(2\theta-1)x - \theta}$
- Higher degree polynomials
- Gordon-Hall



Transformation map for MOR

Examples: θ is the point of maximum height or of steepest solution.

- Translation:
- Dilatation:
- Higher degree polynomials
- Gordon-Hall



Transformation examples

Translation (for periodic BC): $T^{-1}(\mu, x) = x - \theta(\mu, t) + 0.5$

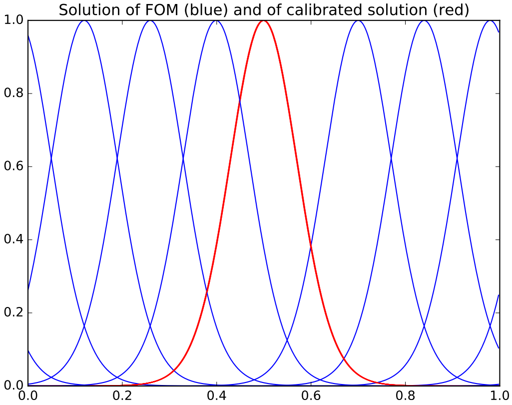


Figure: Calibrated and original solutions for traveling wave

POD of calibrated solutions

Translation (for periodic BC): $T^{-1}(\mu, x) = x - \theta(\mu, t) + 0.5$

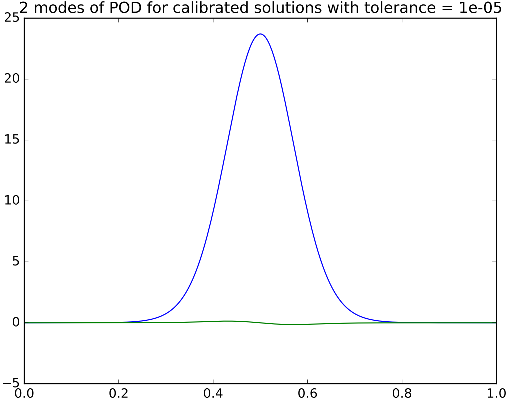


Figure: POD of calibrated solutions for traveling wave

Transformation examples

Dilatation (for other BCs): $T^{-1}(\theta, x) = x\frac{\theta-1}{(2\theta-1)x-\theta}$

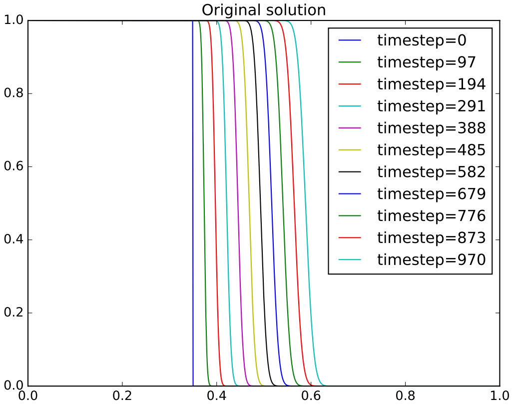


Figure: Original solutions for traveling shock

Transformation examples

Dilatation (for other BCs): $T^{-1}(\theta, x) = x\frac{\theta-1}{(2\theta-1)x-\theta}$

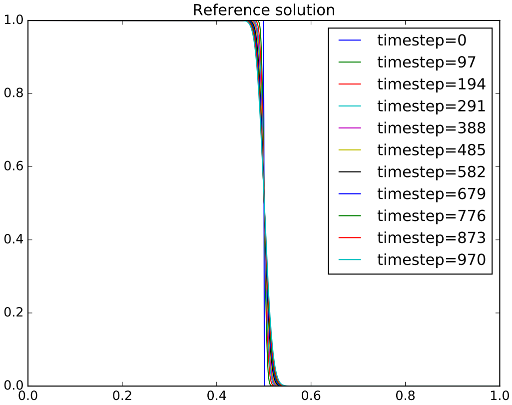


Figure: Calibrated solutions for traveling shock

POD of calibrated solutions

Dilatation (for other BCs): $T^{-1}(\theta, x) = x^{\frac{\theta-1}{(2\theta-1)x-\theta}}$

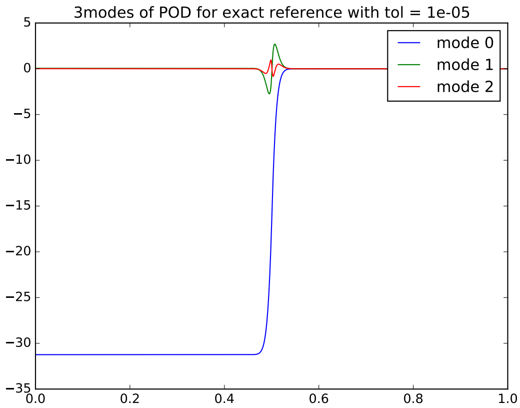


Figure: POD of calibrated solutions for travelling shock

Outline

Arbitrary Lagrangian–Eulerian formulation

$$\frac{d}{dt}u(x,t,\boldsymbol{\mu}) + \frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) = 0$$

$$x := T(\theta(t,\boldsymbol{\mu}),y), \quad v(y,t,\boldsymbol{\mu}) := u(T(\theta(t,\boldsymbol{\mu}),y),t,\mu) = u(x,t,\boldsymbol{\mu})$$

$$\begin{aligned}\frac{d}{dt}v(y,t,\boldsymbol{\mu}) =& \frac{d}{dt}u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})\\
=&\partial_t u(x,t,\boldsymbol{\mu}) + \partial_x u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\
=&-\frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\
=&-\frac{dy}{dx}\frac{d}{dy}F(v(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\end{aligned}$$

Arbitrary Lagrangian–Eulerian formulation

$$\frac{d}{dt}u(x,t,\boldsymbol{\mu}) + \frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) = 0$$

$$x := T(\theta(t,\boldsymbol{\mu}),y), \quad v(y,t,\boldsymbol{\mu}) := u(T(\theta(t,\boldsymbol{\mu}),y),t,\mu) = u(x,t,\boldsymbol{\mu})$$

$$\begin{aligned}\frac{d}{dt}v(y,t,\boldsymbol{\mu}) =& \frac{d}{dt}u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})\\ =& \partial_t u(x,t,\boldsymbol{\mu}) + \partial_x u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\ =& -\frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\ =& -\frac{dy}{dx}\frac{d}{dy}F(v(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\end{aligned}$$

Arbitrary Lagrangian–Eulerian formulation

$$\frac{d}{dt}u(x,t,\boldsymbol{\mu}) + \frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) = 0$$

$$x := T(\theta(t,\boldsymbol{\mu}),y), \quad v(y,t,\boldsymbol{\mu}) := u(T(\theta(t,\boldsymbol{\mu}),y),t,\mu) = u(x,t,\boldsymbol{\mu})$$

$$\begin{aligned}\frac{d}{dt}v(y,t,\boldsymbol{\mu}) =& \frac{d}{dt}u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})\\ =& \partial_t u(x,t,\boldsymbol{\mu}) + \partial_x u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\ =& -\frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\ =& -\frac{dy}{dx}\frac{d}{dy}F(v(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\end{aligned}$$

Arbitrary Lagrangian–Eulerian formulation

$$\frac{d}{dt}u(x,t,\boldsymbol{\mu}) + \frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) = 0$$

$$x := T(\theta(t,\boldsymbol{\mu}),y), \quad v(y,t,\boldsymbol{\mu}) := u(T(\theta(t,\boldsymbol{\mu}),y),t,\mu) = u(x,t,\boldsymbol{\mu})$$

$$\begin{aligned}
\frac{d}{dt}v(y,t,\boldsymbol{\mu}) =& \frac{d}{dt}u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu})\\
=&\partial_t u(x,t,\boldsymbol{\mu}) + \partial_x u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\
=& -\frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}\\
=& -\frac{dy}{dx}\frac{d}{dy}F(v(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}
\end{aligned}$$

Arbitrary Lagrangian–Eulerian formulation

$$\frac{d}{dt}u(x,t,\boldsymbol{\mu}) + \frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) = 0$$

$$x := T(\theta(t,\boldsymbol{\mu}),y), \quad v(y,t,\boldsymbol{\mu}) := u(T(\theta(t,\boldsymbol{\mu}),y),t,\mu) = u(x,t,\boldsymbol{\mu})$$

$$\begin{aligned}
\frac{d}{dt}v(y,t,\boldsymbol{\mu}) =& \frac{d}{dt}u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu}) \\
=& \partial_t u(x,t,\boldsymbol{\mu}) + \partial_x u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} \\
=& -\frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} \\
=& -\frac{dy}{dx}\frac{d}{dy}F(v(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}
\end{aligned}$$

Arbitrary Lagrangian–Eulerian formulation

$$\frac{d}{dt}u(x,t,\boldsymbol{\mu}) + \frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) = 0$$

$$x := T(\theta(t,\boldsymbol{\mu}),y), \quad v(y,t,\boldsymbol{\mu}) := u(T(\theta(t,\boldsymbol{\mu}),y),t,\mu) = u(x,t,\boldsymbol{\mu})$$

$$\begin{aligned}
\frac{d}{dt}v(y,t,\boldsymbol{\mu}) =& \frac{d}{dt}u(T(\theta(t,\boldsymbol{\mu}),y),t,\boldsymbol{\mu}) \\
=& \partial_t u(x,t,\boldsymbol{\mu}) + \partial_x u(x,t,\boldsymbol{\mu}) \frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} \\
=& -\frac{d}{dx}F(u(x,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{d}{dx}u(x,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt} \\
=& -\frac{dy}{dx}\frac{d}{dy}F(v(y,t,\boldsymbol{\mu}),\boldsymbol{\mu}) + \frac{dy}{dx}\frac{d}{dy}v(y,t,\boldsymbol{\mu})\frac{dT(\theta(t,\boldsymbol{\mu}),y)}{dt}
\end{aligned}$$

Arbitrary Lagrangian–Eulerian formulation

$$\frac{\partial}{\partial t}v(y,\boldsymbol{\mu},t)+\frac{dy}{dx}\frac{d}{dy}F(v,\mu)-\frac{dy}{dx}\frac{dv}{dy}\frac{\partial T}{\partial t}=0$$

With ALE formulation we can apply the EIM procedure with points on the reference domain $\mathcal{R}$.

Arbitrary Lagrangian–Eulerian formulation

$$\frac{\partial}{\partial t}v(y,\boldsymbol{\mu},t) + \frac{dy}{dx}\frac{d}{dy}F(v,\mu) - \frac{dy}{dx}\frac{dv}{dy}\frac{\partial T}{\partial t} = 0$$

With ALE formulation we can apply the EIM procedure with points on the reference domain $\mathcal{R}$.

Implication of ALE formulation

- We must know $T(\theta(t, \boldsymbol{\mu}), y)$
  - Offline phase: detect some interesting points (maxima, steepest gradient)
  - Offline phase: optimize the transformation in some sense (T. Taddei, Ohlberger et al.)
  - Online phase: predict the value of the transformation (regression/ machine learning techniques (RNN) / projections)
- Compute the Jacobian of the transformation $\frac{dy}{dx}$ and the new flux $\frac{dv}{dy}$ $\Rightarrow$ increasing computational costs also in online phase

Implication of ALE formulation

- We must know $T(\theta(t, \boldsymbol{\mu}), y)$
  - Offline phase: detect some interesting points (maxima, steepest gradient)
  - Offline phase: optimize the transformation in some sense (T. Taddei, Ohlberger et al.)
  - Online phase: predict the value of the transformation (regression/ machine learning techniques (RNN) / projections)
- Compute the Jacobian of the transformation $\frac{dy}{dx}$ and the new flux $\frac{dv}{dy} \Rightarrow$ increasing computational costs also in online phase

Learning of θ

- Offline: optimization process on a training sample
- Generation of a regression map

Piecewise linear regression for every timestep t^n

- If parameter domain is a grid $\Rightarrow$ Easy, fast
- Non–structured parameter domain $\Rightarrow$ Different algorithms, may be costly
- Precise if $|\mathcal{P}_h| \sim s^P$ with s big enough
- May not catch the nonlinear behavior and produce unreasonable results

Learning of θ

- Offline: optimization process on a training sample
- Generation of a regression map

Polynomial regression

$$\theta(\boldsymbol{\mu}, t) \approx \sum_{|\alpha| \le p} \beta_\alpha t^{\gamma_0} \prod_{i=1}^{p} \mu_i^{\gamma_i} \tag{6}$$

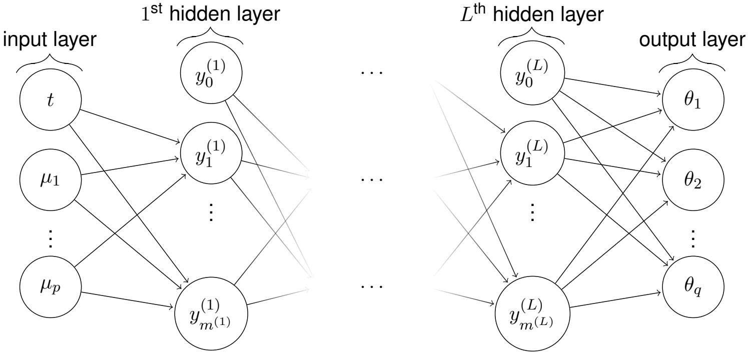
- Hyperparameter p
- Risk of overfitting
- Can easily catch the nonlinear behavior
- Number of coefficients grows exponentially with p

Learning of θ

Neural networks

- Why? Naturally nonlinear, we may not have a structured dictionary
- Which one? Multi-layer-perceptron, recursive neural network (RNN)

Multilayer perceptron



Learning of θ

Neural networks

- Why? Naturally nonlinear, we may not have a structured dictionary
- Which one? Multi-layer-perceptron, recursive neural network (RNN)

Multilayer perceptron

- Many hyperparameters: N layers ($[4, 10]$), M_n nodes ($[6, 20]$)
- Not so precise, error ~ 20 cells

Review of the algorithm

INITIALIZATION:

- Compute or optimize $\theta(\boldsymbol{\mu}_k, t^n)$ for some $\boldsymbol{\mu}_k \in \mathcal{P}$ and $n \leq N_t$
- Build the regression map $\hat{\theta} : \mathcal{P} \times \mathbb{R}^+ \to \mathbb{R}^q$
- EIM on $\tilde{\mathcal{L}}(U^n, \boldsymbol{\mu}_0, t^n, \hat{\theta}(\boldsymbol{\mu}_0, t^n))$ for $n \leq N_t$
- $RB = POD(\{U^n(\boldsymbol{\mu}_0)\}_{n=0}^{N_t})$

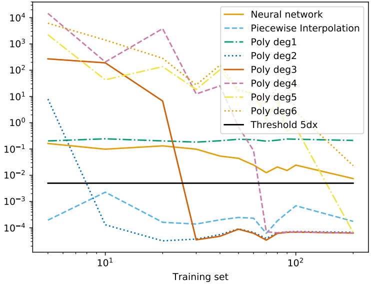
ITERATION:

- Greedy algorithm spanning over the parameter space $\mathcal{P}_h$, with an error indicator $\varepsilon(\mathbf{U}(\boldsymbol{\mu}, t^n, \hat{\theta}(\boldsymbol{\mu}, t^n)))$ where $\mathbf{U} \in \mathbb{R}^N$
- Choose worst parameter as $\boldsymbol{\mu}^* = \arg\max_{\boldsymbol{\mu} \in \mathcal{P}_h} \varepsilon(\mathbf{U}(\boldsymbol{\mu}))$
- Apply POD on time evolution of selected solution $POD_{add} = POD\left(\{U^n(\boldsymbol{\mu}^*)\}_{n=1}^{N_t}\right)$
- Update the RB with $RB = POD\,(RB \cup POD_{add})$
- Update EIM basis function with $EIM_{space} = EIM_{space} \cup EIM(\{\tilde{\mathcal{L}}(U^n, \boldsymbol{\mu}^*, t^n, \hat{\theta}(\boldsymbol{\mu}^*, t^n))\}_{n=0}^{N_t})$

Outline

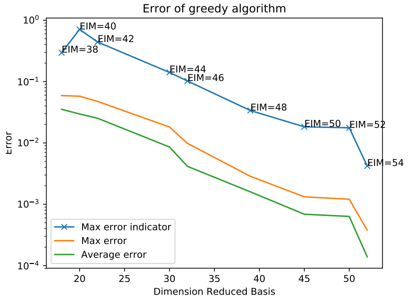
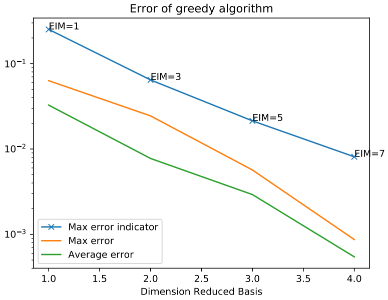
Advection: traveling wave

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = e^{-\mu_1 (x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

Without calibration	With calibration: Regressions
	

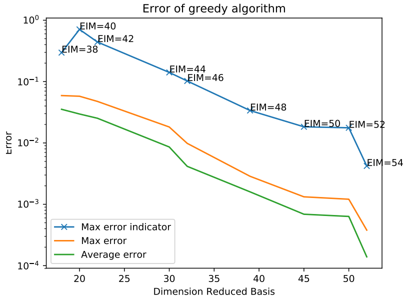
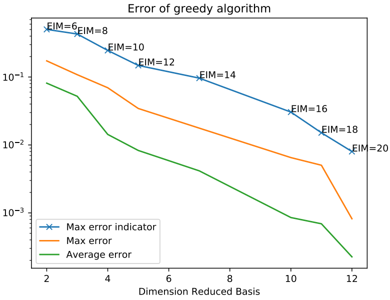
Advection: traveling wave

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = e^{-\mu_1 (x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

Without calibration	With calibration: Poly2
	

Advection: traveling wave

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = e^{-\mu_1 (x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

Without calibration	With calibration: ANN
	

Advection: traveling wave

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{periodic BC} \\ u_0(x,\boldsymbol{\mu}) = e^{-\mu_1 (x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

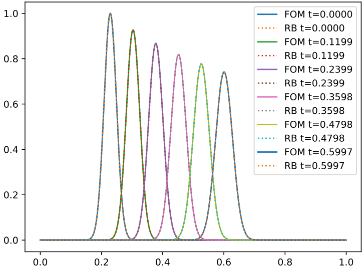
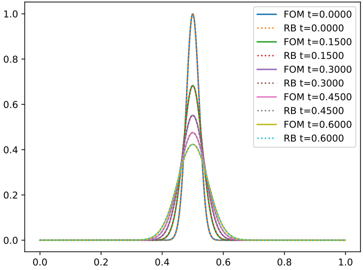
Without calibration		With calibration: Poly2	
RB dim	52	RB dim	4
EIM dim	54	EIM dim	7
FOM time	191 s	FOM time	516 s
RB time	24 s	RB time	18 s
RB/FOM time	12%	RB/FOM time	3%

Advection: traveling wave

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ periodic BC} \\ u_0(x,\boldsymbol{\mu}) = e^{-\mu_1(x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

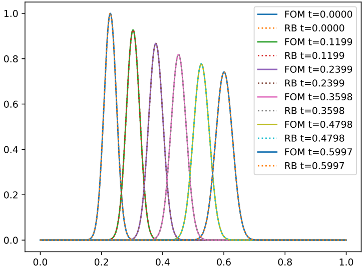
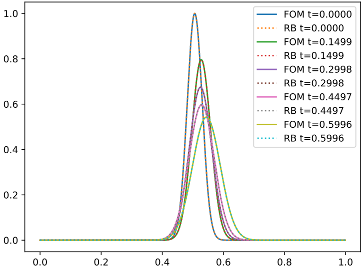
Without calibration		With calibration: ANN	
RB dim	52	RB dim	12
EIM dim	54	EIM dim	20
FOM time	191 s	FOM time	516 s
RB time	24 s	RB time	38 s
RB/FOM time	12%	RB/FOM time	7%

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = e^{-\mu_1 (x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

Without calibration	With calibration: Poly2
	

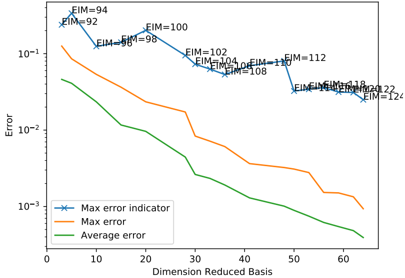
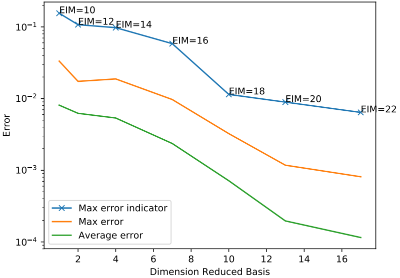
Advection: traveling wave

$$\begin{cases} u_t + \mu_0 u_x = 0, \; D = [0,1], \; T_{max} = 0.6, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = e^{-\mu_1 (x-\mu_2)^2} \\ \mu_0 \sim \mathcal{U}([0,2]), \mu_1 \sim \mathcal{U}([500,1500]), \mu_2 \sim \mathcal{U}([0.1,0.3]) \end{cases}$$

Without calibration	With calibration: ANN
	

Advection: traveling shock

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 1.5, \text{ Dirichlet BC} \\ u_0(x,\boldsymbol{\mu}) = \begin{cases} \mu_1 & \text{if } x < 0.35 + 0.05\mu_2 \\ 0 & \text{else} \end{cases} \\ \mu_0 \sim \mathcal{U}([0,2]),\ \mu_1,\mu_2 \sim \mathcal{U}([-1,1]) \end{cases}$$

Without calibration	With calibration: Poly2
	

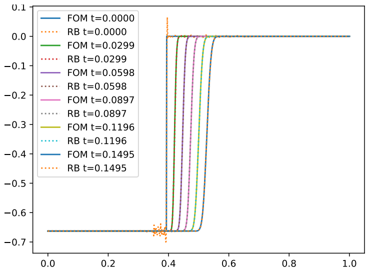
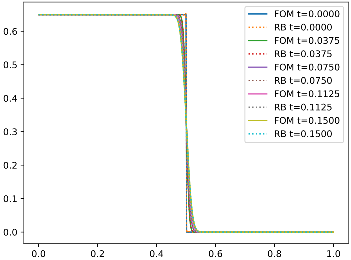
Advection: traveling shock

$$\begin{cases} u_t + \mu_0 u_x = 0,\ D = [0,1],\ T_{max} = 1.5, \text{ Dirichlet BC} \\ u_0(x,\boldsymbol{\mu}) = \begin{cases} \mu_1 & \text{if } x < 0.35 + 0.05\mu_2 \\ 0 & \text{else} \end{cases} \\ \mu_0 \sim \mathcal{U}([0,2]),\ \mu_1, \mu_2 \sim \mathcal{U}([-1,1]) \end{cases}$$

Without calibration		With calibration: Poly2	
RB dim	64	RB dim	17
EIM dim	124	EIM dim	22
FOM time	49 s	FOM time	125 s
RB time	9 s	RB time	6 s
RB/FOM time	18%	RB/FOM time	5%

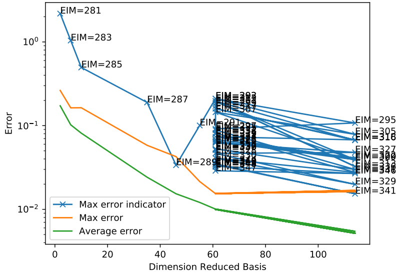
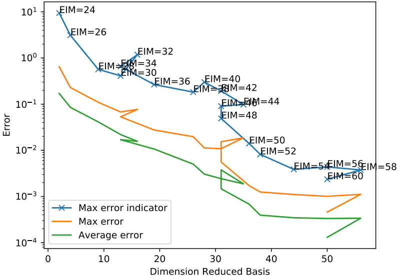
Advection: traveling shock

$$\begin{cases} u_t + \mu_0 u_x = 0, \; D = [0,1], \; T_{max} = 1.5, \text{ Dirichlet BC} \\ u_0(x, \boldsymbol{\mu}) = \begin{cases} \mu_1 & \text{if } x < 0.35 + 0.05\mu_2 \\ 0 & \text{else} \end{cases} \\ \mu_0 \sim \mathcal{U}([0,2]), \; \mu_1, \mu_2 \sim \mathcal{U}([-1,1]) \end{cases}$$

Without calibration	With calibration: Poly2
	

Burgers oscillation

$$\begin{cases} u_t + \mu_0 (u^2/2)_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ Dirichlet BC} \\ u_0(x,\boldsymbol{\mu}) = \sin(2\pi(x+0.1\mu_1))e^{-(60+20\mu_2)(x-0.5)^2}(1+0.5\mu_3 x) \\ \mu_0 \sim \mathcal{U}([0,2]),\ \mu_1 \sim \mathcal{U}([0,1]),\ \mu_2, \mu_3 \sim \mathcal{U}([-1,1]) \end{cases}$$

Without calibration	With calibration: Poly3
	

Burgers oscillation

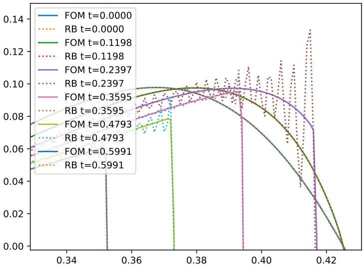
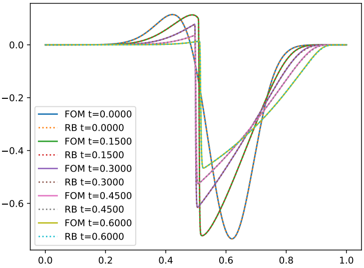
$$\begin{cases} u_t + \mu_0 (u^2/2)_x = 0,\ D = [0,1],\ T_{max} = 0.6, \text{ Dirichlet BC} \\ u_0(x,\boldsymbol{\mu}) = \sin(2\pi(x+0.1\mu_1))e^{-(60+20\mu_2)(x-0.5)^2}(1+0.5\mu_3 x) \\ \mu_0 \sim \mathcal{U}([0,2]),\ \mu_1 \sim \mathcal{U}([0,1]),\ \mu_2,\mu_3 \sim \mathcal{U}([-1,1]) \end{cases}$$

Without calibration[2]		With calibration: Poly3	
RB dim	153	RB dim	50
EIM dim	335	EIM dim	60
FOM time	119 s	FOM time	314 s
RB time	50 s	RB time	35 s
RB/FOM time	42%	RB/FOM time	11%

[2] It does not reach the requested tolerance 10^{-3}

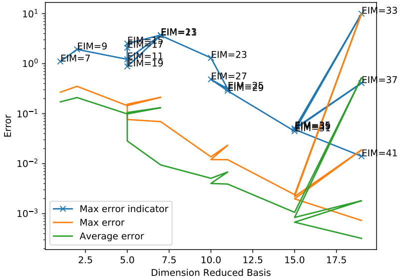
Burgers oscillation

$$\begin{cases} u_t + \mu_0 (u^2/2)_x = 0,\; D = [0,1],\; T_{max} = 0.6, \text{ Dirichlet BC} \\ u_0(x, \boldsymbol{\mu}) = \sin(2\pi(x + 0.1\mu_1)) e^{-(60+20\mu_2)(x-0.5)^2} (1 + 0.5\mu_3 x) \\ \mu_0 \sim \mathcal{U}([0,2]),\; \mu_1 \sim \mathcal{U}([0,1]),\; \mu_2, \mu_3 \sim \mathcal{U}([-1,1]) \end{cases}$$

Without calibration	With calibration: Poly3
	

Burgers sine

$$\begin{cases} u_t + \mu_0 (u^2/2)_x = 0, \; D = [0,\pi], \; T_{max} = 0.15, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = |\sin(x+\mu_1)| + 0.1 \\ \mu_0 \sim \mathcal{U}([0,2]), \;\; \mu_1 \sim \mathcal{U}([0,\pi]) \end{cases}$$

Without calibration	With calibration: Poly3
	

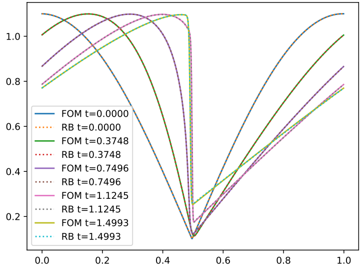
Burgers sine

$$\begin{cases} u_t + \mu_0 (u^2/2)_x = 0,\ D=[0,\pi],\ T_{max}=0.15, \text{ periodic BC} \\ u_0(x,\boldsymbol{\mu}) = |\sin(x+\mu_1)| + 0.1 \\ \mu_0 \sim \mathcal{U}([0,2]),\ \mu_1 \sim \mathcal{U}([0,\pi]) \end{cases}$$

Without calibration		With calibration: Poly3	
RB dim	failed	RB dim	19
EIM dim	>600	EIM dim	41
FOM time	167 s	FOM time	444 s
RB time	∞	RB time	53 s
RB/FOM time	∞	RB/FOM time	11%

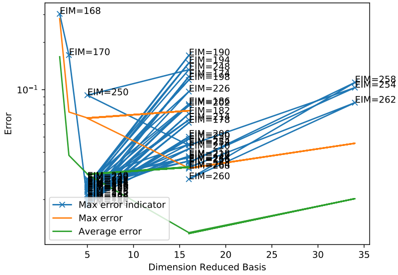
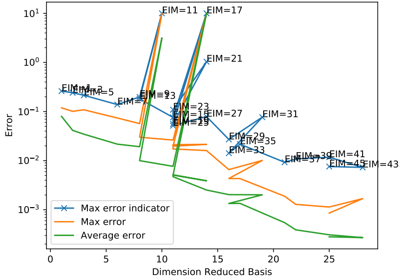
Burgers sine

$$\begin{cases} u_t + \mu_0 (u^2/2)_x = 0, \; D = [0,\pi], \; T_{max} = 0.15, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = |\sin(x+\mu_1)| + 0.1 \\ \mu_0 \sim \mathcal{U}([0,2]), \; \mu_1 \sim \mathcal{U}([0,\pi]) \end{cases}$$

Without calibration	With calibration: Poly3
	

Buckley-Leverett equation

$$\begin{cases} \partial_t u + \partial_x \dfrac{u^2}{u^2 + \mu_0(1-u^2)} = 0,\ D = [0,1],\ T_{max} = 0.25, \text{ periodic BC} \\ u_0(x,\boldsymbol{\mu}) = 0.5 + 0.2\mu_1 + 0.3\mu_1 \sin(2\pi(x-\mu_1-0.5)) \\ \mu_0 \sim \mathcal{U}([0.001,2]),\ \mu_1 \sim \mathcal{U}([0.1,1]) \end{cases}$$

Without calibration	With calibration: pwL
	

Buckley-Leverett equation

$$\begin{cases} \partial_t u + \partial_x \dfrac{u^2}{u^2+\mu_0(1-u^2)} = 0,\ D=[0,1],\ T_{max}=0.25, \text{ periodic BC} \\ u_0(x,\boldsymbol{\mu}) = 0.5 + 0.2\mu_1 + 0.3\mu_1 \sin(2\pi(x-\mu_1-0.5)) \\ \mu_0 \sim \mathcal{U}([0.001,2]),\ \mu_1 \sim \mathcal{U}([0.1,1]) \end{cases}$$

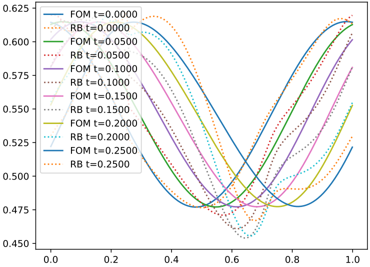
Without calibration[2]		With calibration: pwL	
RB dim	16	RB dim	25
EIM dim	270	EIM dim	45
FOM time	190 s	FOM time	462 s
RB time	69 s	RB time	79 s
RB/FOM time	36%	RB/FOM time	17%

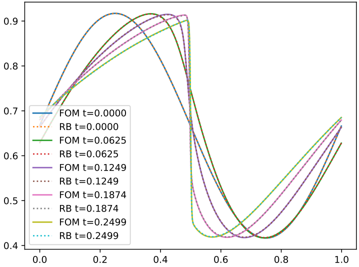
[2] It does not reach the requested tolerance 10^{-3}

Buckley-Leverett equation

$$\begin{cases} \partial_t u + \partial_x \dfrac{u^2}{u^2 + \mu_0(1-u^2)} = 0,\ D = [0,1],\ T_{max} = 0.25, \text{ periodic BC} \\ u_0(x, \boldsymbol{\mu}) = 0.5 + 0.2\mu_1 + 0.3\mu_1 \sin(2\pi(x - \mu_1 - 0.5)) \\ \mu_0 \sim \mathcal{U}([0.001, 2]),\ \mu_1 \sim \mathcal{U}([0.1, 1]) \end{cases}$$

Without calibration	With calibration: pwL





Outline

Extensions and limitations

Extensions

- More dimensions
- Systems of equations
- Non crossing multiple shocks or waves

Limitations

- Crossing shocks or waves and Riemann's problem for systems of equations
  - Problem: ideal transformation becomes degenerate when two shocks collide
  - Solution: remeshing/ categorization and use of different RB spaces
- More D transformation can be challenging to be described with few parameters

Perspectives

Perspectives

- Extend the algorithm to systems and 2D tests
- Extend the algorithm to more complicated transformations for many non–crossing shocks
- Developing a RNN to predict the θ values in the online phase

Search me: Google Scholar, arXiv (preprint)

Thank you for the attention!

Residual Distribution

- High order
- FE based
- Compact stencil
- Explicit
- Can recast some other FV, FE, FD, DG schemes[2]

$$\partial_t U + \nabla \cdot F(U) = 0 \tag{7}$$

$$V_h = \{U \in L^2(\Omega_h, \mathbb{R}^D) \cap \mathcal{C}^0(\Omega_h),\, U|_K \in \mathbb{P}^k,\, \forall K \in \Omega_h\}. \tag{8}$$

$$U_h = \sum_{\sigma \in D_{\mathcal{N}}} U_\sigma \varphi_\sigma = \sum_{K \in \Omega_h} \sum_{\sigma \in K} U_\sigma \varphi_\sigma|_K \tag{9}$$

[2] **R. Abgrall. Computational Methods in Applied Mathematics; 2018.**

Residual Distribution

- High order
- FE based
- Compact stencil
- Explicit
- Can recast some other FV, FE, FD, DG schemes[2]

$$\partial_t U + \nabla \cdot F(U) = 0 \tag{7}$$

$$V_h = \{U \in L^2(\Omega_h, \mathbb{R}^D) \cap \mathcal{C}^0(\Omega_h),\, U|_K \in \mathbb{P}^k,\, \forall K \in \Omega_h\}. \tag{8}$$

$$U_h = \sum_{\sigma \in D_\mathcal{N}} U_\sigma \varphi_\sigma = \sum_{K \in \Omega_h} \sum_{\sigma \in K} U_\sigma \varphi_\sigma|_K \tag{9}$$

[2] **R. Abgrall. Computational Methods in Applied Mathematics; 2018.**

Residual Distribution - Spatial Discretization

1. Define $\forall K \in \Omega_h$ a fluctuation term (total residual) $\phi^K = \int_K \nabla \cdot F(U) dx$
2. Define a nodal residual $\phi^K_\sigma \; \forall \sigma \in K$:

$$\phi^K = \sum_{\sigma \in K} \phi^K_\sigma, \quad \forall K \in \Omega_h. \tag{10}$$

3. The resulting scheme is

$$U^{n+1}_\sigma - U^n_\sigma + \Delta t \sum_{K|\sigma \in K} \phi^K_\sigma = 0, \quad \forall \sigma \in D_{\mathcal{N}}. \tag{11}$$

Residual Distribution

- High order
- Easy to code
- FE based
- Compact stencil
- No need of Riemann solver
- No need of conservative variables
- Can recast some other FV, FE schemes

$$\partial_t U + \nabla \cdot A(U) = S(U) \tag{12}$$

$$V_h = \{U \in L^2(\Omega_h, \mathbb{R}^D) \cap \mathcal{C}^0(\Omega_h),\, U|_K \in \mathbb{P}^k,\, \forall K \in \Omega_h\}. \tag{13}$$

$$U_h = \sum_{\sigma \in D_\mathcal{N}} U_\sigma \varphi_\sigma = \sum_{K \in \Omega_h} \sum_{\sigma \in K} U_\sigma \varphi_\sigma|_K \tag{14}$$

Residual Distribution

- High order
- Easy to code
- FE based
- Compact stencil
- No need of Riemann solver
- No need of conservative variables
- Can recast some other FV, FE schemes

$$\partial_t U + \nabla \cdot A(U) = S(U) \tag{12}$$

$$V_h = \{U \in L^2(\Omega_h, \mathbb{R}^D) \cap \mathcal{C}^0(\Omega_h),\, U|_K \in \mathbb{P}^k,\, \forall K \in \Omega_h\}. \tag{13}$$

$$U_h = \sum_{\sigma \in D_{\mathcal{N}}} U_\sigma \varphi_\sigma = \sum_{K \in \Omega_h} \sum_{\sigma \in K} U_\sigma \varphi_\sigma|_K \tag{14}$$

Residual Distribution - Spatial Discretization

Focus on steady case.

1. Define $\forall K \in \Omega_h$ a fluctuation term (total residual) $\phi^K = \int_K \nabla \cdot A(U) - S(U) dx$
2. Define a nodal residual $\phi_\sigma^K \; \forall \sigma \in K$:

$$\phi^K = \sum_{\sigma \in K} \phi_\sigma^K, \quad \forall K \in \Omega_h. \tag{15}$$

   Often done assigning $\phi_\sigma^K = \beta_\sigma^K \phi^K$, where must hold that

$$\sum_{\sigma \in K} \beta_\sigma^K = \mathrm{Id}. \tag{16}$$

3. The resulting scheme is

$$\sum_{K|\sigma \in K} \phi_\sigma^K = 0, \quad \forall \sigma \in D_{\mathcal{N}}. \tag{17}$$

   This will be called residual distribution scheme.

Residual distribution - Choice of the scheme

How to split total residuals into nodal residuals $\Rightarrow$ choice of the scheme.

$$\phi_\sigma^{K,LxF}(U_h) = \int_K \varphi_\sigma \left(\nabla \cdot A(U_h) - S(U_h)\right) dx + \alpha_K (U_\sigma - \overline{U}_h^K),$$

$$\overline{U}_h^K = \int_K U_h, \quad \alpha_K = \max_{e \text{ edge } \in K} \left(\rho_S \left(\nabla A(U_h) \cdot \mathbf{n}_e \right) \right),$$

$$\beta_\sigma^K(U_h) = \max\left(\frac{\Phi_\sigma^{K,LxF}}{\Phi^K}, 0 \right) \left(\sum_{j \in K} \max\left(\frac{\Phi_j^{K,LxF}}{\Phi^K}, 0 \right) \right)^{-1}, \tag{18}$$

$$\phi_\sigma^{*,K} = (1-\Theta)\beta_\sigma^K \phi_\sigma^K + \Theta \Phi_\sigma^{K,LxF}, \quad \Theta = \frac{|\Phi^K|}{\sum_{j\in K} |\Phi_j^{K,LxF}|},$$

$$\phi_\sigma^K = \beta_\sigma^K \phi_\sigma^{*,K} + \sum_{e | \text{edge of } K} \theta h_e^2 \int_e [\nabla U_h] \cdot [\nabla \varphi_\sigma] d\Gamma.$$

Error estimator

Additional hypothesis:

- $Id + \Delta t \mathcal{L}$ is Liptschitz continuous with constant $C > 0$,
- There are N'_{EIM} extra functions and functionals that capture the evolution of the solutions. (experimentally not so strict),
- Initial conditions are exactly represented in the reduced basis RB.

Total error estimator:

- EIM error, estimated by other N'_{EIM} basis functions f and functional τ iterating the EIM procedure after the stop, cost $\mathcal{O}(N'_{EIM})$,
- RB error given by the Lipschitz constant times residual of the small system,
- additionally one can add the projection error of the initial condition when not in RB.

Empirical interpolation method (EIM)

INPUT: $\mathcal{L}^n(U^n, \boldsymbol{\mu}, t^n)$, for $\boldsymbol{\mu} \in \mathcal{P}_h$, $n \leq N_t$

OUTPUT: $EIM = (\tau_k, f_k)_{k=1}^{N_{EIM}}$ where functions $f_k \in \mathbb{R}^{\mathcal{N}}$ and $\tau_k \in (\mathbb{R}^{\mathcal{N}})'$ (Examples of τ_k are point evaluations)

- Greedy iterative procedure
- At each step chooses the worst approximated function via an error estimator $\mathcal{L}^{worst} = \arg\max_{\mathcal{L}} ||\mathcal{L} - \sum_{k=1}^{N_{EIM}} \tau_k(\mathcal{L}) f_k||$
- Maximise the functional τ on the function $\mathcal{L}^{worst}$ $\tau^{chosen} = \arg\max_{\tau} |\tau(\mathcal{L}^{worst})|$
- $EIM = EIM \cup (\tau^{chosen}, \mathcal{L}^{worst})$
- Stop when error is smaller than a tolerance

Proper orthogonal decomposition (POD)

INPUT: Collection of functions $\{f_j\}_{j=1}^N$

OUTPUT: Reduced basis spaces

$$RB = \underset{U|dim(U)=N_{POD}}{\arg\min} \sum_{j=1}^N ||f_j - \mathcal{P}_U(f_j)||_2$$

- Based on SVD
- Prescribed tolerance to stop the algorithm
- Global optimizer of the problem

Greedy algorithm

INPUT: Collection of functions $\{f_j\}_{j=1}^N$

OUTPUT: Reduced basis space RB

- There is an error estimator (normally cheap) $\varepsilon_{RB}(f) \sim ||f - \mathcal{P}_{RB}(f)||$
- Iteratively choose the worst represented function $f^{worst} = \arg\max_f \varepsilon_{RB}(f)$
- Add f^{worst} to the RB space
- Stop up to a certain tolerance