Discovery mathematical models from experimental data

Alessandro Alla

Seminario di Modellistica Differenziale Numerica
Sapienza, Universita' di Roma

Rome, December 18, 2018

Outline

1. Dynamic Mode Decomposition and its applications
   - Dynamic Mode Decomposition
   - Reduction of PDEs (with J.N. Kutz)
   - Human Mobility (with C. Balzotti, M. Briani, E. Cristiani)

2. Discovery Nonlinear PDEs (with. S. Rudy, S. Brunton, J.N. Kutz)
   - Identification of constant coefficient PDEs
   - Identification of parametric PDEs

Outline

Dynamic Mode Decomposition (Schmid, '12)

Dynamic Mode Decomposition

Suppose we have a dynamical system and compute snapshots $\{(y(t_0), \ldots, y(t_m)\}$ and two sets of data

$$\mathbf{Y} = \begin{bmatrix} | & | & & | \\ \mathbf{y}(t_0) & \mathbf{y}(t_1) & \cdots & \mathbf{y}(t_{m-1}) \\ | & | & & | \end{bmatrix}, \quad \mathbf{Y}' = \begin{bmatrix} | & | & & | \\ \mathbf{y}(t_1) & \mathbf{y}(t_2) & \cdots & \mathbf{y}(t_m) \\ | & | & & | \end{bmatrix}$$

with $\mathbf{y}(t_j)$ an initial condition of the dynamical system and $\mathbf{y}(t_{j+1})$ its corresponding output $\Rightarrow \mathbf{Y}' = \mathbf{A_Y Y}$ with $\mathbf{A_Y} \in \mathbb{R}^{n \times n}$ unknown

The DMD modes are eigenvectors of

$$\mathbf{A_y} = \mathbf{Y}'\mathbf{Y}^{\dagger}$$

where $\dagger$ denotes the Moore-Penrose pseudoinverse

Dynamic Mode Decomposition (Schmid, '12)

DMD produces a regression procedure whereby the data snapshots in time are used to produce the **best-fit linear dynamical system** for **Y**. DMD procedure constructs approximate linear evolution

$$\frac{d\widetilde{\mathbf{y}}}{dt} = \hat{\mathbf{A}}_{\mathbf{y}}\widetilde{\mathbf{y}}$$

with $\widetilde{\mathbf{y}}(0) = \widetilde{\mathbf{y}}_0$ and whose solution is

$$\widetilde{\mathbf{y}}(t) = \sum_{i=1}^{n} b_i \psi_i \exp(\omega_i t)$$

ψ_i and ω_i are the eigenfunctions and eigenvalues of the matrix $\hat{\mathbf{A}}_{\mathbf{y}}$

Dynamic Mode Decomposition

DMD allows to build this approximation system:

$$\mathbf{M}\dot{\mathbf{y}}(t) = \mathbf{A}\mathbf{y}(t) + \mathbf{f}(t, \mathbf{y}(t)) \approx \mathbf{A}\mathbf{y}(t) + \hat{\mathbf{A}}_{\mathbf{y}}\mathbf{y}(t)$$

$\hat{\mathbf{A}}_{\mathbf{y}}$ approximates the nonlinearity over the snapshots collected

PROBLEM

The matrix $\hat{\mathbf{A}}_{\mathbf{y}}$ is, highly **ill-conditioned** and when the state dimension n is large can be even **intractable** to analyze directly

SOLUTION

DMD circumvents the eigendecomposition of $\hat{\mathbf{A}}_{\mathbf{y}}$ by considering a rank-reduced representation in terms of a POD-projected matrix $\widetilde{\mathbf{A}}_{\mathbf{y}}$

DMD algorithm

Require: Snapshots $\{\mathbf{y}(t_0), \ldots, \mathbf{y}(t_m)\}$

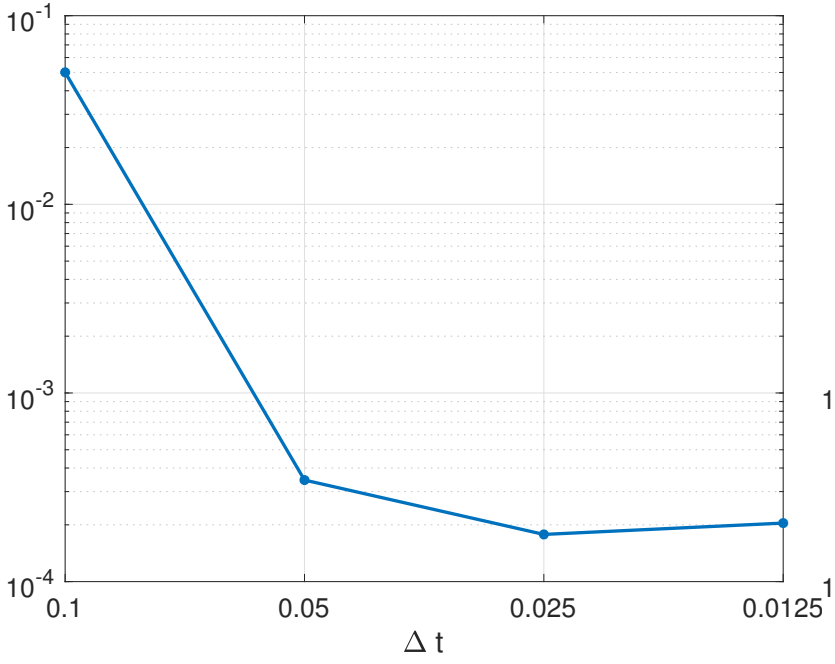
1: Set $\mathbf{Y} = [\mathbf{y}(t_0), \ldots, \mathbf{y}(t_{m-1})]$ and $\mathbf{Y}' = [\mathbf{y}(t_1), \ldots, \mathbf{y}(t_m)]$
2: Compute the **(reduced)** SVD of $\mathbf{Y}$, $\mathbf{Y} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$
3: Define $\widetilde{\mathbf{A}}_\mathbf{y} := \mathbf{U}^*\mathbf{Y}'\mathbf{V}\mathbf{\Sigma}^{-1}$
4: Compute eigenvalues and eigenvectors of $\widetilde{\mathbf{A}}_\mathbf{y}\mathbf{W} = \mathbf{W}\mathbf{\Lambda}$
5: Set $\mathbf{\Psi}^{\text{DMD}} = \mathbf{Y}'\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{W}$

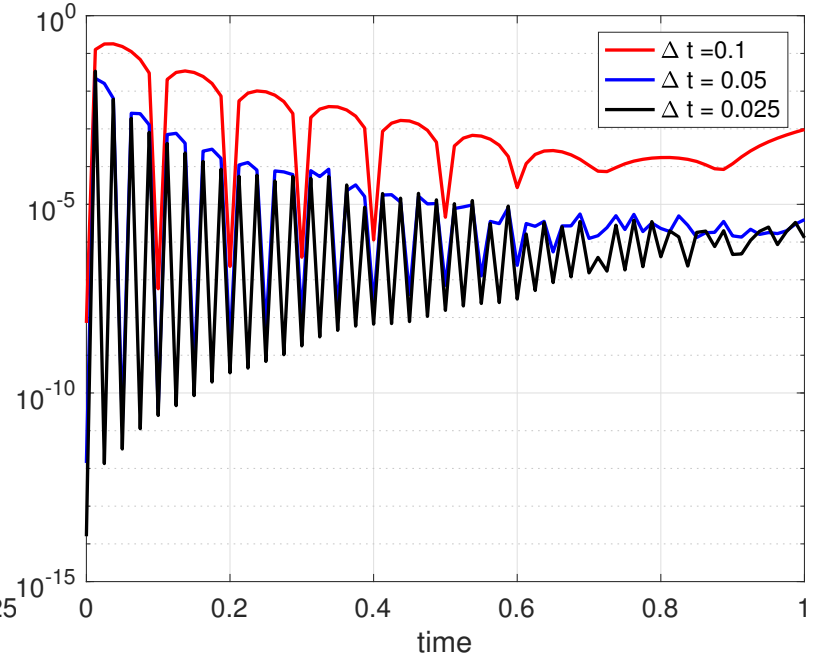
2D Viscous Burgers

$$y_t - \frac{1}{100}\Delta y + y \cdot \nabla y = 0, \quad x \in \Omega = [0,1] \times [0,1], t \in [0,T]$$

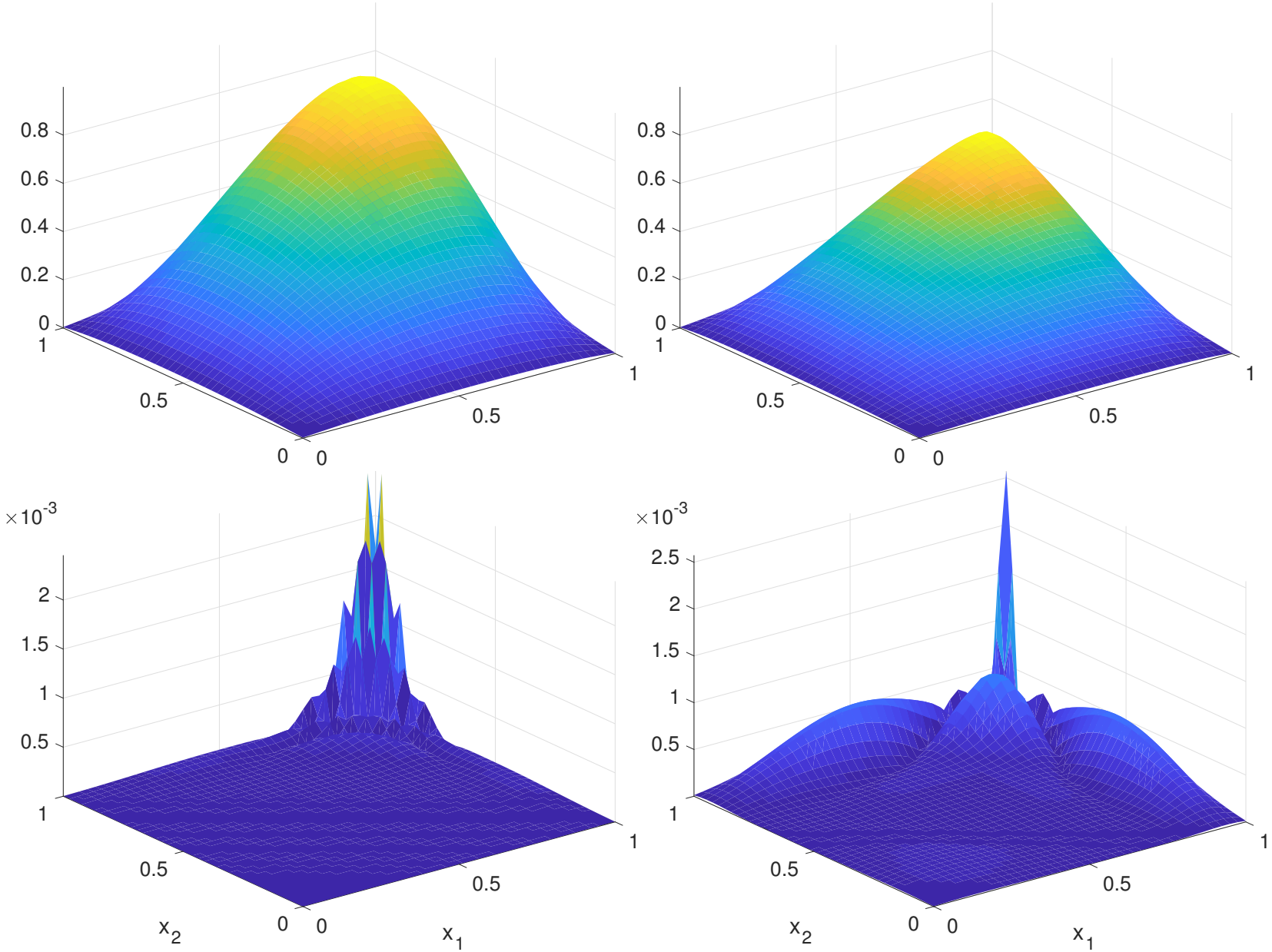
$$y(x,t) = 0, \quad x \in \partial\Omega$$

$$y(x,0) = \sin(\pi x) \quad x \in \Omega$$





2D Viscous Burgers



Problem Settings

Dynamical System

$$\begin{cases} \mathbf{M}\dot{\mathbf{y}}(t) = \mathbf{A}\mathbf{y}(t) + \mathbf{f}(t, \mathbf{y}(t)), \quad t \in (0, T] \\ \\ \mathbf{y}(0) = \mathbf{y_0} \end{cases}$$

Assumptions

- $\mathbf{y_0} \in \mathbb{R}^n$ is a given initial data
- $\mathbf{M}, \mathbf{A} \in \mathbb{R}^{n \times n}$ given matrices
- $\mathbf{f} : [0, T] \times \mathbb{R}^n \to \mathbb{R}^n$ a continuous function in both arguments and locally Lipschitz-type with respect to the second variable

WARNING: High dimensional problems are computationally expensive

Proper Orthogonal Decomposition and SVD

Given **snapshots** $(y(t_0), \ldots, y(t_n)) \in \mathbb{R}^m$

We look for an orthonormal basis $\{\psi_i\}_{i=1}^{\ell}$ in $\mathbb{R}^m$ with $\ell \ll \min\{n, m\}$ s.t.

$$J(\psi_1, \ldots, \psi_\ell) = \sum_{j=1}^{n} \alpha_j \left\| y_j - \sum_{i=1}^{\ell} \langle y_j, \psi_i \rangle \psi_i \right\|^2 = \sum_{i=\ell+1}^{d} \sigma_i^2$$

reaches a minimum where $\{\alpha_j\}_{j=1}^{n} \in \mathbb{R}^+$.

$$\min J(\psi_1, \ldots, \psi_\ell) \quad s.t. \langle \psi_i, \psi_j \rangle = \delta_{ij}$$

Singular Value Decomposition: $Y = \Psi \Sigma V^T$

For $\ell \in \{1, \ldots, d = rank(Y)\}$, $\{\psi_i\}_{i=1}^{\ell}$ are called **POD basis** of rank ℓ

ERROR INDICATOR: $\mathcal{E}(\ell) = \dfrac{\sum_{i=1}^{\ell} \sigma_i}{\sum_{i=1}^{d} \sigma_i}$ with σ_i singular values of the SVD

Reduced Order System

POD-Galerkin ansatz

$$\mathbf{y}(t) \approx \boldsymbol{\Psi}^{\text{POD}} \mathbf{y}^{\ell}(t), \quad \boldsymbol{\Psi}^{\text{POD}} \in \mathbb{R}^{n \times \ell}$$

POD dynamical system

$$\begin{cases} \mathbf{M}^{\ell} \dot{\mathbf{y}}^{\ell}(t) = \mathbf{A}^{\ell} \mathbf{y}^{\ell}(t) + (\boldsymbol{\Psi}^{\text{POD}})^{T} f(t, \boldsymbol{\Psi}^{\text{POD}} \mathbf{y}^{\ell}(t)) \\ \mathbf{y}^{\ell}(0) = \mathbf{y}_{\mathbf{0}}^{\ell} \end{cases}$$

Dimension of the entries

- $(\mathbf{M}^{\ell})_{ij} = \langle \mathbf{M}\psi_i, \psi_j \rangle \in \mathbb{R}^{\ell \times \ell}$
- $(\mathbf{A}^{\ell})_{ij} = \langle \mathbf{A}\psi_i, \psi_j \rangle \in \mathbb{R}^{\ell \times \ell}$
- $\mathbf{y}_{\mathbf{0}}^{\ell} = (\boldsymbol{\Psi}^{\text{POD}})^{T} \mathbf{y}_0 \in \mathbb{R}^{\ell}$

POD-DMD method (A., Kutz, '17)

MAIN IDEA

The evaluation of the nonlinearity is the **most expensive** part in model order reduction. We aim faster approximation of the nonlinear term

We need snapshots!

- $\{\mathbf{y}(t_0), \ldots, \mathbf{y}(t_m)\}$, to compute the POD basis functions
- $\{\mathbf{f}(t_0, \mathbf{y}(t_0)), \ldots, \mathbf{f}(t_m, \mathbf{y}(t_m))\}$ to compute the DMD basis functions

Compact notation

$$\widetilde{\mathbf{f}}^{\mathrm{DMD}}(t, y(t)) \approx \boldsymbol{\Psi}^{\mathrm{DMD}} diag(e^{\omega^{\mathrm{DMD}} t})\mathbf{b}$$

- $\mathbf{b} = (\boldsymbol{\Psi}^{\mathrm{DMD}})^{\dagger}\mathbf{f}(t_1, \mathbf{y}(t_1)) \in \mathbb{R}^k$
- $diag(e^{\omega^{\mathrm{DMD}} t})b \in \mathbb{R}^k$ reduced approximation wrt DMD modes

Reduced Order Dynamics

POD method

$$\begin{cases} \mathbf{M}^{\ell}\dot{\mathbf{y}}^{\ell}(t) = \mathbf{A}^{\ell}\mathbf{y}^{\ell}(t) + (\boldsymbol{\Psi}^{\mathrm{POD}})^{T} f(t, \boldsymbol{\Psi}^{\mathrm{POD}}\mathbf{y}^{\ell}(t)) \\ \mathbf{y}^{\ell}(0) = \mathbf{y}_{\mathbf{0}}^{\ell} \end{cases}$$

POD-DEIM method (low-rank approximation of the nonlinear term)

$$\begin{cases} \mathbf{M}^{\ell}\dot{\mathbf{y}}^{\ell}(t) = \mathbf{A}^{\ell}\mathbf{y}^{\ell}(t) + (\boldsymbol{\Psi}^{\mathrm{POD}})^{T}\boldsymbol{\Psi}^{\mathrm{DEIM}}\mathbf{f}(t, \mathbf{y}^{\mathrm{DEIM}}) \\ \mathbf{y}^{\ell}(0) = \mathbf{y_0}^{\ell} \end{cases}$$

POD-DMD method **(NO EVALUATION OF THE NONLINEARITY)**

$$\begin{cases} \mathbf{M}^{\ell}\dot{\mathbf{y}}^{\ell}(t) = \mathbf{A}^{\ell}\mathbf{y}^{\ell}(t) + (\boldsymbol{\Psi}^{\mathrm{POD}})^{T}\boldsymbol{\Psi}^{\mathrm{DMD}} diag(e^{\omega^{\mathrm{DMD}}t})\mathbf{b} \\ \mathbf{y}^{\ell}(0) = \mathbf{y_0}^{\ell} \end{cases}$$

Example: Semi-Linear Parabolic Equation

$$
\begin{aligned}
y_t - \theta \Delta y + \mu(y - y^3) = 0 \qquad & (x,t) \in \Omega \times [0,T] \\
y(x,0) = y_0(x) \qquad & x \in \Omega \\
y(\cdot,t) = 0 \qquad & x \in \partial\Omega, t \in [0,T]
\end{aligned}
$$

Parameters:

$\Omega = [0,1] \times [0,1]$, $T = 3$
$y_0(x) = 0.1$ if $0.1 \leq x_1 x_2 \leq 0.6$ and 0 elsewhere

Example: Semi-Linear Parabolic Equation

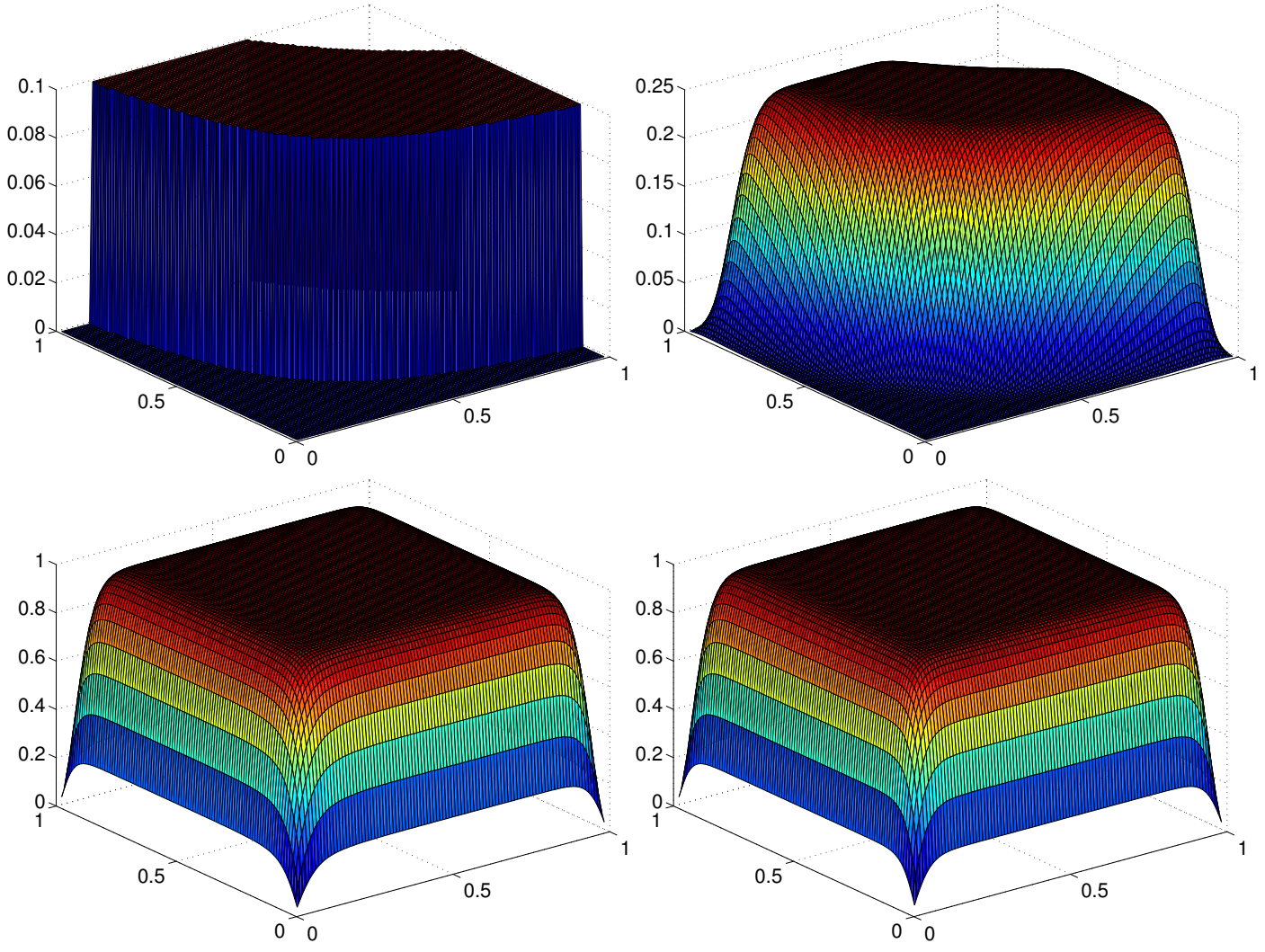
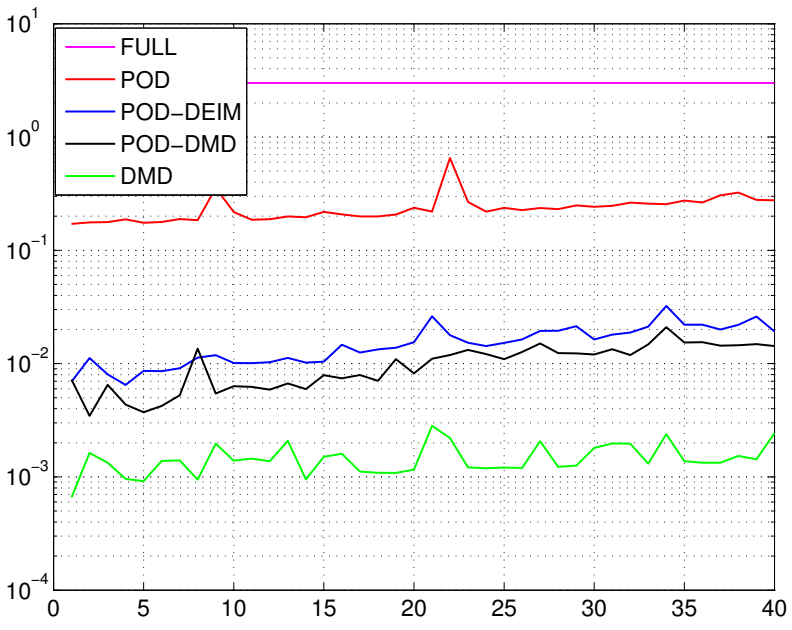


Figure: Solution at time $t = \{0, 0.1\}$ (top) and $t = \{1.5, 3\}$ (bottom)

Example: Semi-Linear Parabolic Equation



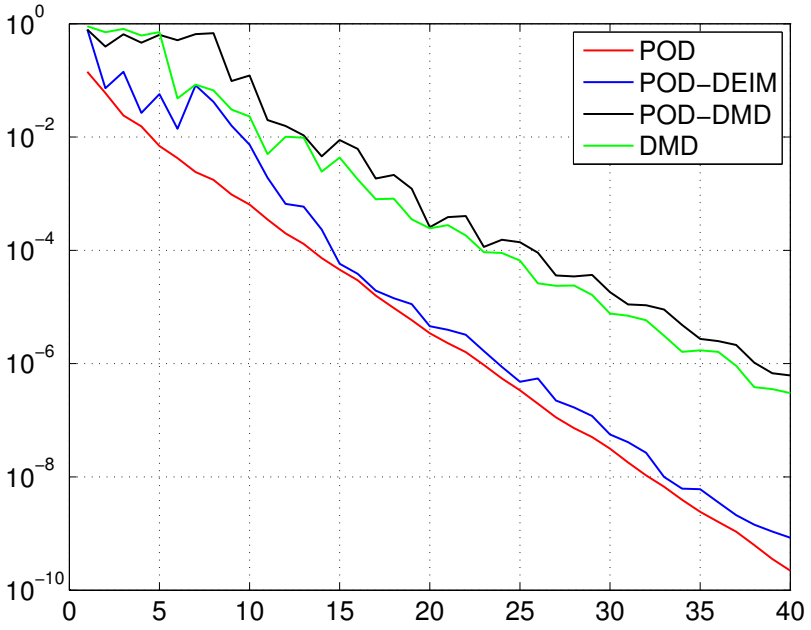
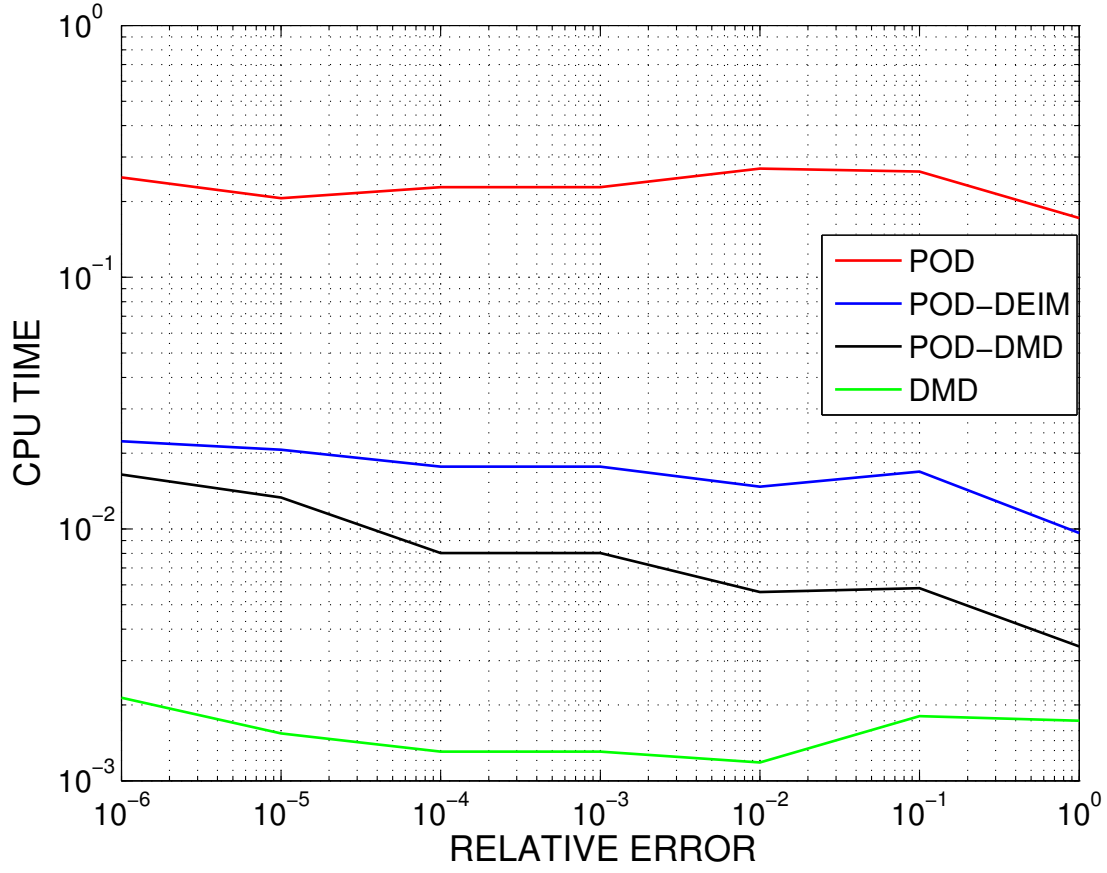


Figure: CPU-time online stage (left) and Relative Error wrt Frobenius norm. Number of POD modes and DEIM/DMD points are the same

Example: Semi-Linear Parabolic Equation

A fair comparison



Human Mobility

Dataset

- The Italian telecommunication company TIM provided the data
- The data are estimates of mobile phones presence in a given area
- The area under analysis is the province of Milan (Italy), divided in $511 \times 389 = 198{,}779$ cells
- We have six months of data, divided into time intervals of 15 minutes, therefore we have 96 data per day per cell

First analyses on data

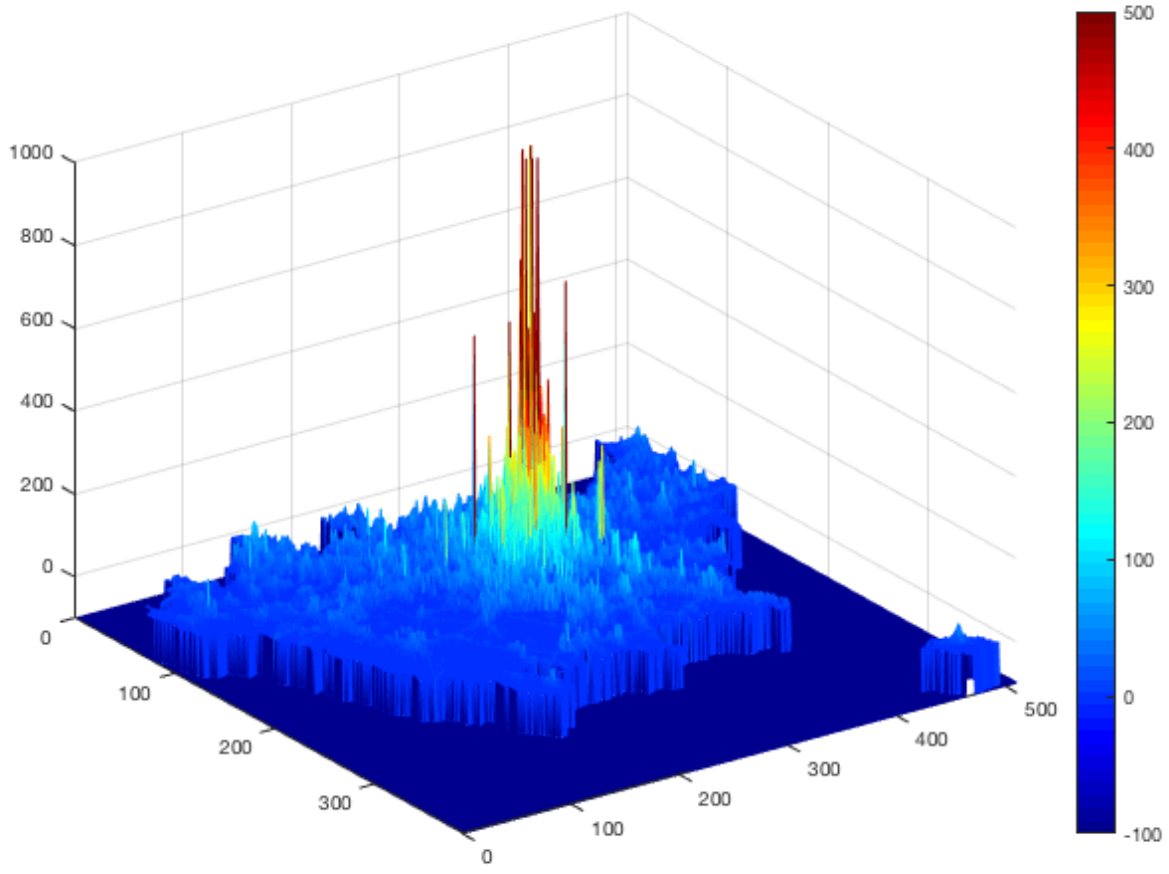
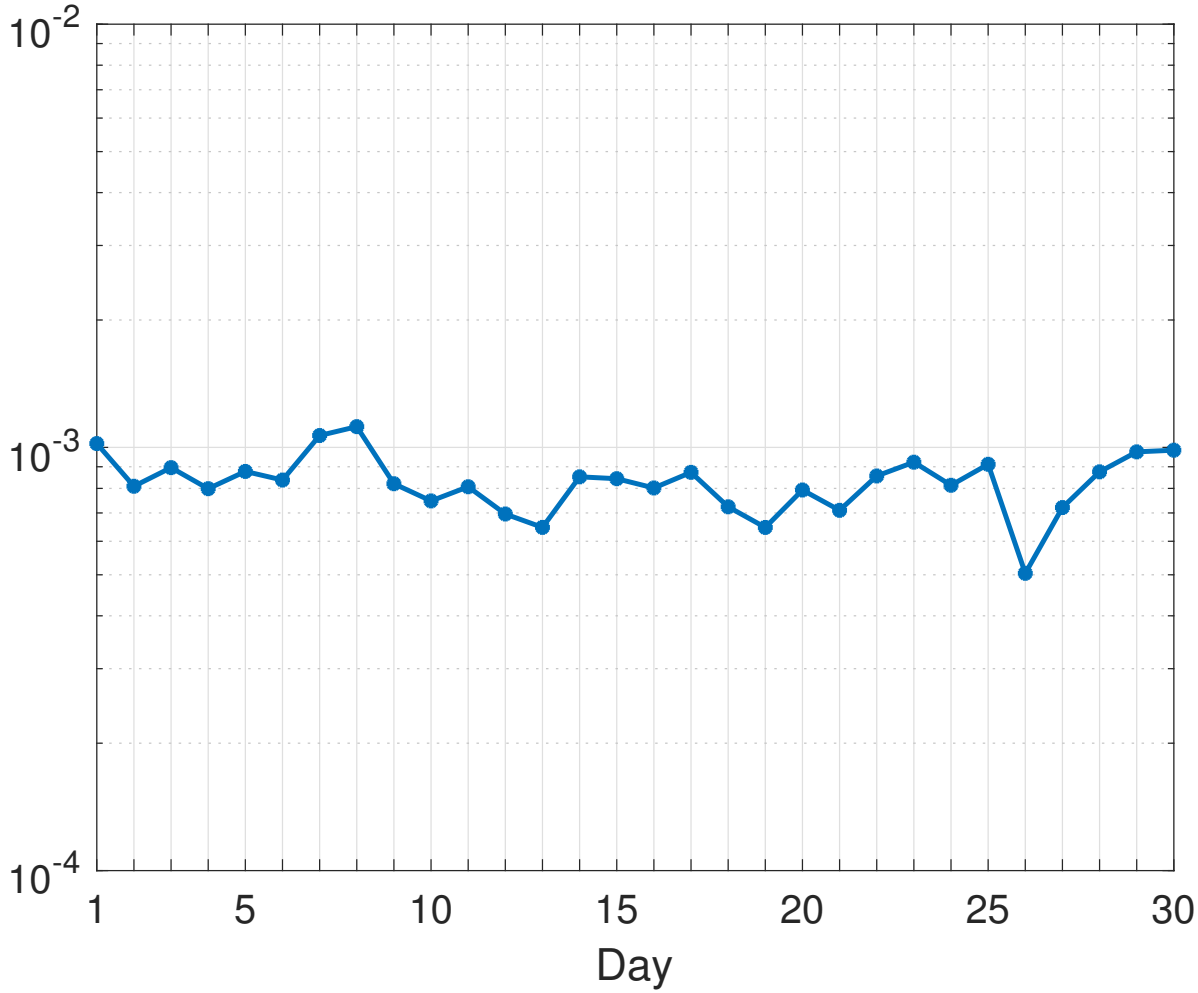


Figure: 3D-plot of the number of TIM users in each cell of Milan's province on April 18, 2017 at 10:00 a.m.

DMD model

DMD error on the TIM data



Monge-Kantorovich Problem

Monge–Kantorovich mass transfer problem

Given a sandpile with mass distribution ρ_0 and a pit with equal volume and mass distribution ρ_1, find a way to minimize the cost of transporting sand into the pit

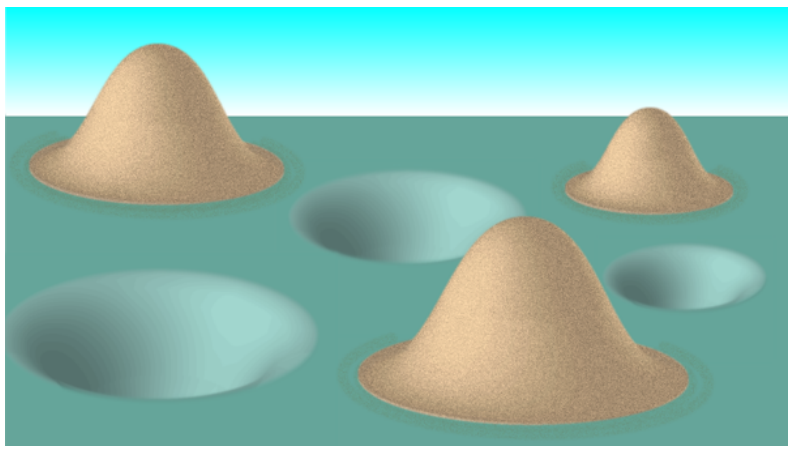


Figure: Sand to be moved into the pits

The cost for moving mass depends on:

- the distance from the point of origin to the point of arrival
- the amount of mass to be moved

Wasserstein Distance

Wasserstein Distance: Definition in $\mathbb{R}^n$

In the space $\mathbb{R}^n$ equipped with the euclidean metrics, let ρ^0 and ρ^1 be two density functions such that $\int_{\mathbb{R}^n} \rho^0 = \int_{\mathbb{R}^n} \rho^1$. For all $p \in [1, +\infty)$, the L^p-Wasserstein distance between ρ^0 and ρ^1 is

$$W_p(\rho^0, \rho^1) = \left(\min_{T \in \mathcal{T}} \int_{\mathbb{R}^n} \| T(x) - x \|^p_{\mathbb{R}^n} \rho^0(x) dx \right)^{\frac{1}{p}}$$

where

$$\mathcal{T} := \left\{ T : \mathbb{R}^n \to \mathbb{R}^n : \int_B \rho^1(x) dx = \int_{\{x : T(x) \in B\}} \rho^0(x) dx, \forall B \subset \mathbb{R}^n \text{ bounded} \right\}$$

$\mathcal{T}$ is the set of all possible maps which transfer the mass from one configuration to the other

Approximation of Wasserstein Distance

We reformulate the mass transfer problem on a graph $\mathcal{G}$ with N nodes.
We denote by:

- m_j^0 the initial mass distributed on the graph nodes
- m_j^1 the final mass distributed on the graph nodes
- c_{jk} the cost to transfer a unit mass from node j to node k
- x_{jk} the (unknown) mass moving from node j to node k

Rearranging in an optimal manner the first mass in the second one

Minimize

$$\mathcal{H} := \sum_{j,k=1}^{N} c_{jk} x_{jk}$$

subject to

$$\sum_k x_{jk} = m_j^0 \quad \forall j \qquad \sum_j x_{jk} = m_k^1 \quad \forall k \qquad \text{and} \qquad x_{jk} \geq 0$$

Approximation of Wasserstein Distance

We define

$$x = (x_{11}, x_{12}, \dots, x_{1N}, x_{21}, \dots, x_{2N}, \dots, x_{N1}, \dots, x_{NN})^T$$
$$c = (c_{11}, c_{12}, \dots, c_{1N}, c_{21}, \dots, c_{2N}, \dots, c_{N1}, \dots, c_{NN})^T$$
$$b = (m_1^0, \dots, m_N^0, m_1^1, \dots, m_N^1)^T$$

and the matrix

$$A = \begin{bmatrix} \mathbb{1}_N & 0 & 0 & \dots & 0 \\ 0 & \mathbb{1}_N & 0 & \dots & 0 \\ 0 & 0 & \mathbb{1}_N & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \mathbb{1}_N \\ I_N & I_N & I_N & I_N & I_N \end{bmatrix}$$

where I_N is the $N \times N$ identity matrix and $\mathbb{1}_N = \underbrace{(1\ 1 \dots\ 1)}_{N \text{ times}}$

Approximation of Wasserstein Distance

Linear Programming problem

Minimize $c^T x$, under the conditions $Ax = b$ and $x \geq 0$.
The result of the algorithm is a vector $x^* := \arg\min c^T x$ whose elements x^*_{jk} represent how much mass moves from node j to node k

Application of the Algorithm to Human Mobility Flows

- The nodes of the graph $\mathcal{G}$ are the centers of the cells of the province of Milan. Each node is connected to all the others.
- The amount of people located in each cell j represents the mass m_j to be moved.
- The initial mass m^0 and the final one m^1 represent presence data on two consecutive quarters of an hour.
- We assume that people can move in any direction of the space neglecting obstacles and that they aim at minimizing the total displacement as a whole.

Idea

Solving the LP problem with two consecutive mass distributions m^0 and m^1, we get the optimal path followed by people to move from the first configuration to the second one.

Input and Output

- Input: the pairs (m^0, m^1) corresponding to the number of people at two consecutive quarters of an hour.
- Output: x^* solution of the LP problem between two consecutive quarters of an hour.
- We represent the movements with arrows and we draw only those larger than the daily average.
- We analyze small areas or we aggregate the cells into groups to reduce the computational cost.

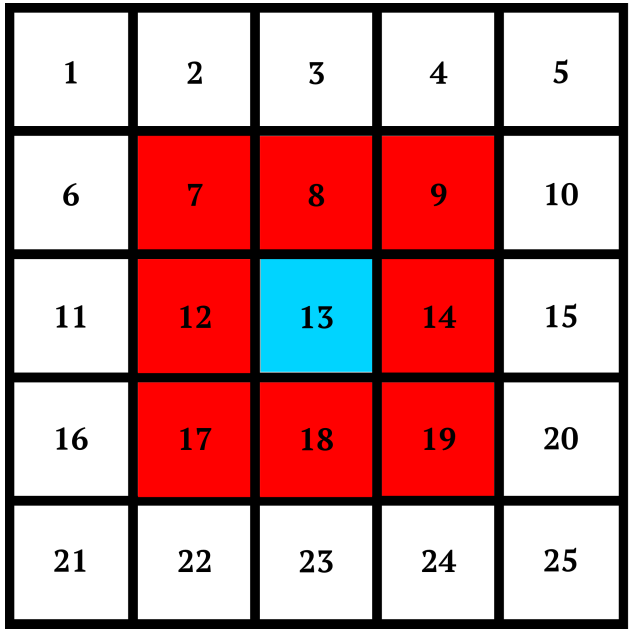
Why DMD?

Problem

LP helps us to understand the flow which moves the mass from every single couple of nodes $\Longrightarrow$ **Computationally Expensive**

Solution

DMD model allows to choose Δt small enough, **to impose the mass on a generic node to move only towards** d **adjacent cells** or not



Reduced Linear Programming problem

Reduced Linear Programming problem

Minimize $\widetilde{c}^T\widetilde{x}$, under the conditions $\widetilde{A}\widetilde{x} = b$ and $\widetilde{x} \geq 0$,
with $|\widetilde{x}| = |\widetilde{c}| \leq dN$, $|b| = 2N$ and $|\widetilde{A}| \leq 2dN^2$

Algoritm	**Vectors Dimension**	**Matrix Dimension**
Global	N^2	$2N^3$
Local	dN	$2dN^2$

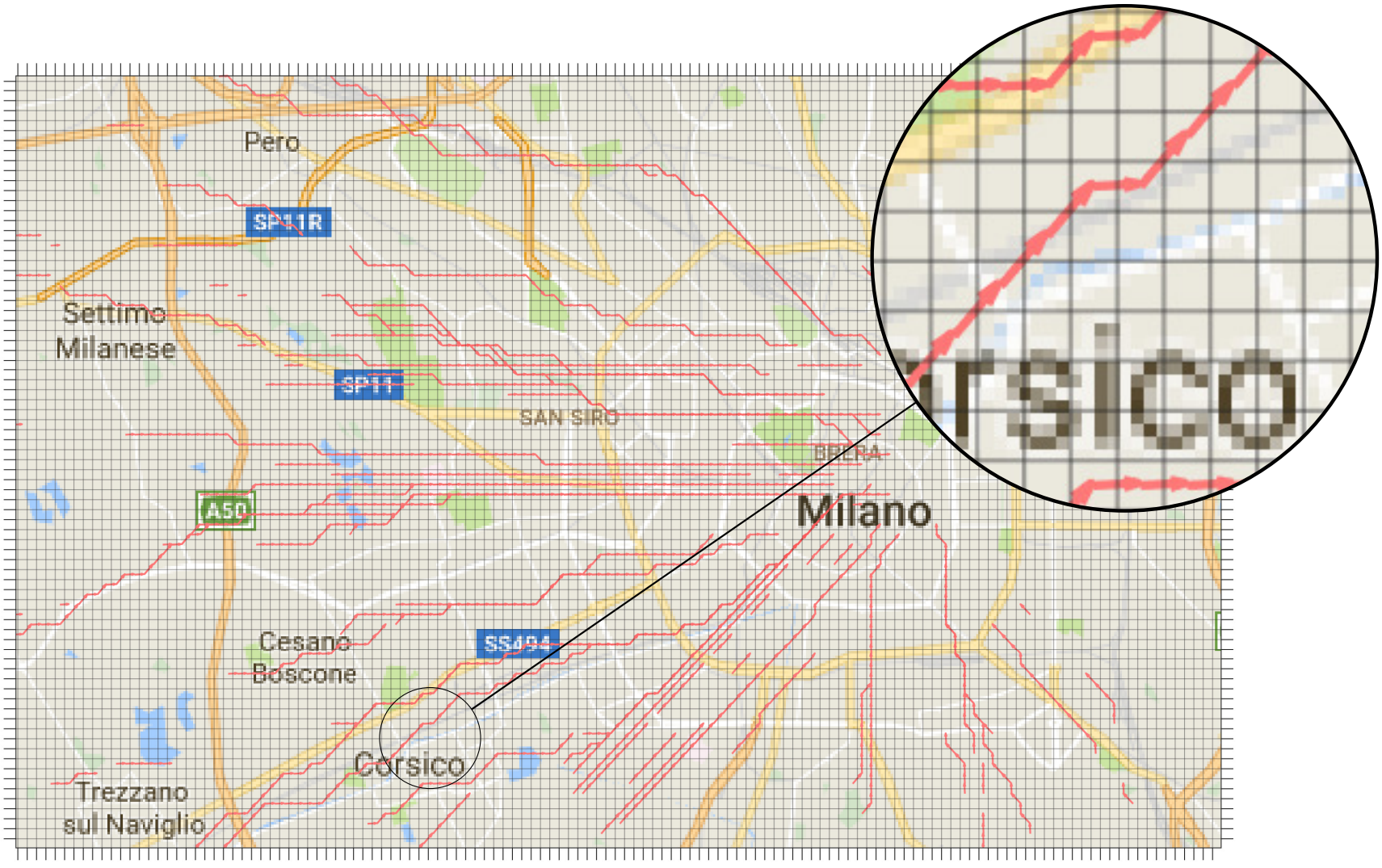


Figure: Flows of commuters during the morning of a generic working day.

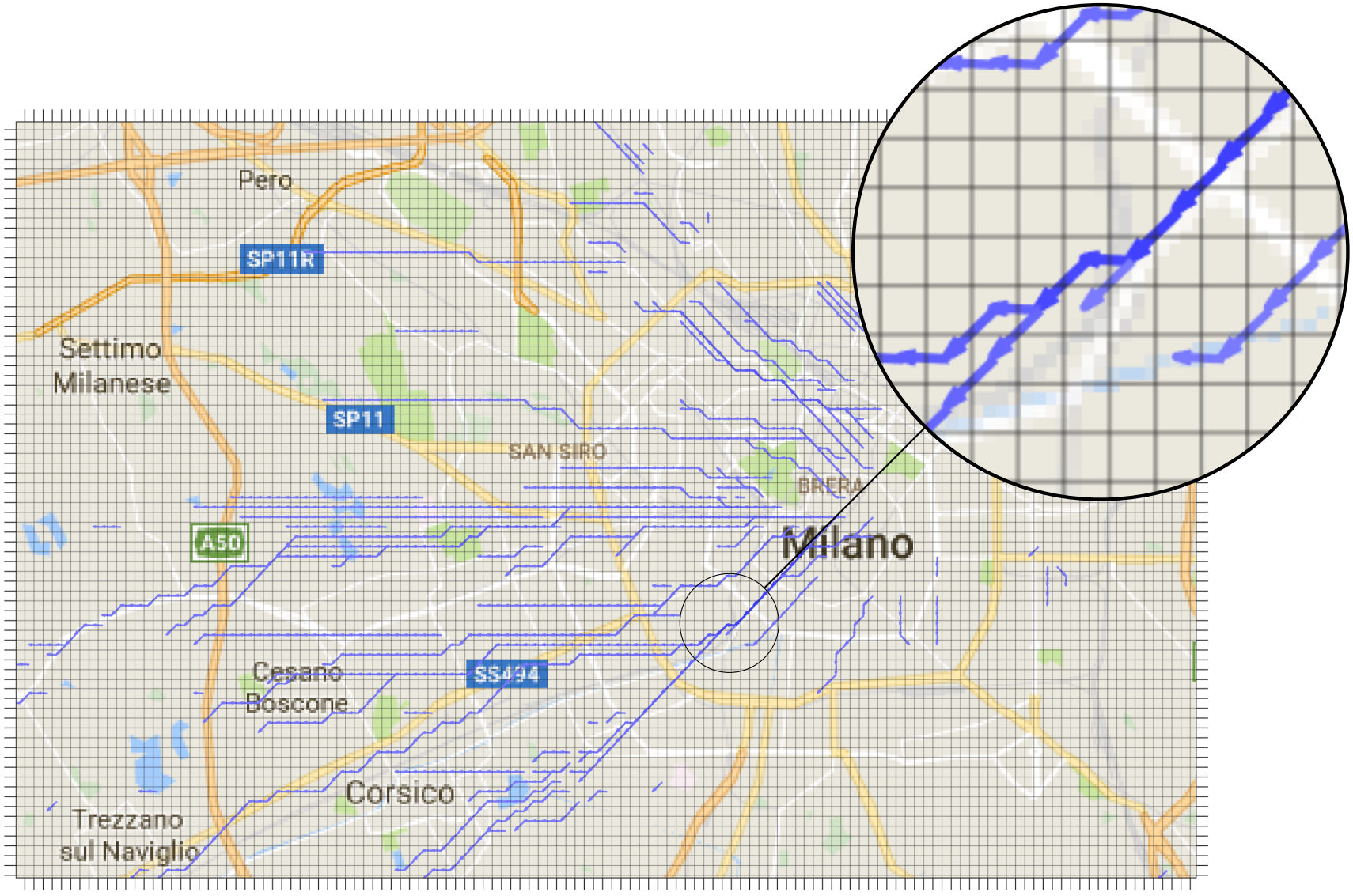


Figure: Flows of commuters during the evening of a generic working day.

Outline

Goal

so far...

we have computed a **linear** model through DMD

now...

we want to extract physical laws from experimental data in the form of PDEs using machine learning methods:

$$u_t = N(u, u_x, u_{xx}, \ldots, \mu(\mathbf{t})) \quad t \in [0, T]$$

$N(\cdot)$: characterizes the evolution of the system

$\mu(t) : [0, T] \to \mathbb{R}$ its parametric dependencies

Identification of PDEs

To identify constant coefficient PDEs, we have a dataset **U**, which is a discretization of a function $u(x,t)$ that we assume satisfies the PDE of the form given in:

$$u_t = N(u, u_x, u_{xx}, \dots) = \sum_{j=1}^{d} N_j(u, u_x, u_{xx}, \dots)\xi_j$$

Assumptions

- Nonlinear expression $N(\cdot)$ may be expanded as a sum of simple monomial basis functions N_j of u and its derivatives
- We build a (complete) library of many possible monomial basis functions and regresses to find ξ
- Sparsity is used to ensure that basis functions that do not appear in the PDE are set to zero in the sum

Identification of PDEs

Given a dataset $\mathbf{U} \in \mathbb{R}^{n \times m}$ representing m timesteps of a PDE discretized with n gridpoints, we numerically differentiate in both x and t to form the linear regression problem given by

$$\underbrace{\begin{pmatrix} u_t(x_1,t_1) \\ u_t(x_2,t_1) \\ \vdots \\ u_t(x_n,t_m) \end{pmatrix}}_{\mathbf{U}_t} = \underbrace{\begin{pmatrix} 1 & u(x_1,t_1) & u_x(x_1,t_1) & \dots & u^3 u_{xxx}(x_1,t_1) \\ 1 & u(x_2,t_1) & u_x(x_2,t_1) & \dots & u^3 u_{xxx}(x_2,t_1) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & u(x_n,t_m) & u_x(x_n,t_m) & \dots & u^3 u_{xxx}(x_n,t_m) \end{pmatrix}}_{\Theta(\mathbf{U})} \xi$$

which is a large, overdetermined linear system of equations $\mathbf{Ax} = \mathbf{b}$

Library

We have shown derivatives up to third order are multiplied by powers of u up to cubic order, but one could include arbitrarily **many library functions.** Solving for ξ and ensuring sparsity gives the PDE.

Sparse Optimization

LASSO (**L**east **A**bsolute **S**hrinkage and **S**election **O**perator)

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \mathbf{A}\mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_1^2$$

Ridge regression

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \mathbf{A}\mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2$$

Reference

- S.L. Brunton, J.L. Proctor, J.N. Kutz. *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, 2016
- S.H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz. *Data-driven discovery of partial differential equations,* 2017

Identification of parametric PDEs

In the identification of parametric PDEs, we consider equations as

$$u_t = N(u, u_x, \ldots, \mu(t)) = \sum_{j=1}^{d} N_j(u, u_x, \ldots)\xi_j(t)$$

Some more details

- To capture spatial variation in the coefficients, we simply replace $\xi(t)$ with $\xi(x)$
- Determine which coefficients are nonzero and find the values of the coefficients for each ξ_j at each timestep (**group sparsity**)
- Prior knowledge of an appropriate set of basis functions may be helpful. In what follows we assume the true dynamics N to lie in the span of our candidate functions $\{N_j\}_{j=1}^{d}$

Mathematical formulation

$$u_t^{(j)} = \Theta\left(u^{(j)}\right)\xi^{(j)},\ j = 1, \ldots, m.$$

where

$$\Theta\left(u^{(j)}\right) = \begin{pmatrix} | & | & | & | \\ 1 & u^{(j)} & \ldots & u^3 u_{xxx}^{(j)} \\ | & | & | & | \end{pmatrix}$$

$$\begin{pmatrix} u_t^{(1)} \\ u_t^{(2)} \\ \vdots \\ u_t^{(m)} \end{pmatrix} = \underbrace{\begin{pmatrix} \Theta\left(u^{(1)}\right) & & & \\ & \Theta\left(u^{(2)}\right) & & \\ & & \ddots & \\ & & & \Theta\left(u^{(m)}\right) \end{pmatrix}}_{\widetilde{\Theta}} \begin{pmatrix} \xi^{(1)} \\ \xi^{(2)} \\ \vdots \\ \xi^{(m)} \end{pmatrix}.$$

Identification of parametric PDEs

We use the notion of group sparsity to find time series representing each parameter in the PDE, rather than single values.

Group LASSO

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{w}} \left\| \mathbf{b} - \sum_{g \in \mathcal{G}} \mathbf{A}^{(g)} \mathbf{w}^{(g)} \right\|_2^2 + \lambda \sum_{g \in \mathcal{G}} \|\mathbf{w}^{(g)}\|_1^2$$

Here $\mathcal{G}$ is a collection of groups, each of which contains a subset of the indices of the snapshots

Group Ridge

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{w}} \left\| \mathbf{b} - \sum_{g \in \mathcal{G}} \mathbf{A}^{(g)} \mathbf{w}^{(g)} \right\|_2^2 + \lambda \sum_{g \in \mathcal{G}} \|\mathbf{w}^{(g)}\|_2^2$$

Sequential thresholding method on ridge regression (for group sparisty)

Algorithm 1: SGTR($\mathbf{A}, \mathbf{b}, \mathcal{G}, \lambda, \epsilon$, maxit, $f(\mathbf{x}) = \|\mathbf{x}\|_2$)

Solves $\mathbf{x} \approx \mathbf{A}^{-1}\mathbf{b}$ with sparsity imposed on groups in $\mathcal{G}$
Initialize coefficients with ridge regression
$\mathbf{x} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \mathbf{A}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$

Threshold groups with small f and repeat
for $iter = 1, \ldots,$ maxit:
    # Remove groups with sufficiently small $f(\mathbf{x}^{(g)})$
    $\mathcal{G} = \{g \in \mathcal{G} : f(\mathbf{x}^{(g)}) > \epsilon\}$
    # Refit these groups (note this sets $\mathbf{x}^{(g)} = 0$ for $g \notin \mathcal{G}$)
    $\mathbf{x} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \sum_{g\in\mathcal{G}} \mathbf{A}^{(g)}\mathbf{w}^{(g)}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$

Get unbiased estimates of coefficients after finding sparsity
$\mathbf{x}^{(\mathcal{G})} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \sum_{g\in\mathcal{G}} \mathbf{A}^{(g)}\mathbf{w}^{(g)}\|_2^2$
return $\mathbf{x}$

Sequential thresholding method on ridge regression (for group sparisty)

Remarks

- Throughout the training, $\mathcal{G}$ tracks the groups that have nonzero coefficients, and it is paired down as we threshold coefficients with sufficiently small relevance, as measured by f
- For m timesteps and d candidate functions in the library, groups are defined as $\mathcal{G} = \{j + d \cdot i : i = 0, \ldots, m-1 : j = 1, \ldots, d\}$
- We use the 2-norm of the coefficients in each group for f but one could also consider arbitrary functions
- We normalize each column of $\mathbf{A}$ and $\mathbf{b}$ so that differences in scale between the groups do not affect the result of the algorithm

Numerical Tests

- We tested the method after introducing white noise with mean magnitude equal to 1% of the L^2-norm of the dataset.
- Noise is added directly to the data, $\mathbf{U}$ prior to numerical differentiation in order to replicate the effects of sensor noise.
- A comparison with group LASSO regression is also given for a number of the examples.

Viscous Burgers' Equation

$$u_t = a(t)uu_x + 0.1u_{xx}, x \in [-8, 8], t \in [0, 10]$$

$$a(t) = -\left(1 + \frac{\sin(t)}{4}\right)$$

Parameters

Library: powers of u up to cubic order, which can be multiplied by derivatives of u up to fourth order

Noise-free dataset we use the discrete Fourier transform for computing derivatives

Noisy dataset, we use polynomial interpolation to smooth the derivatives

Viscous Burgers' Equation

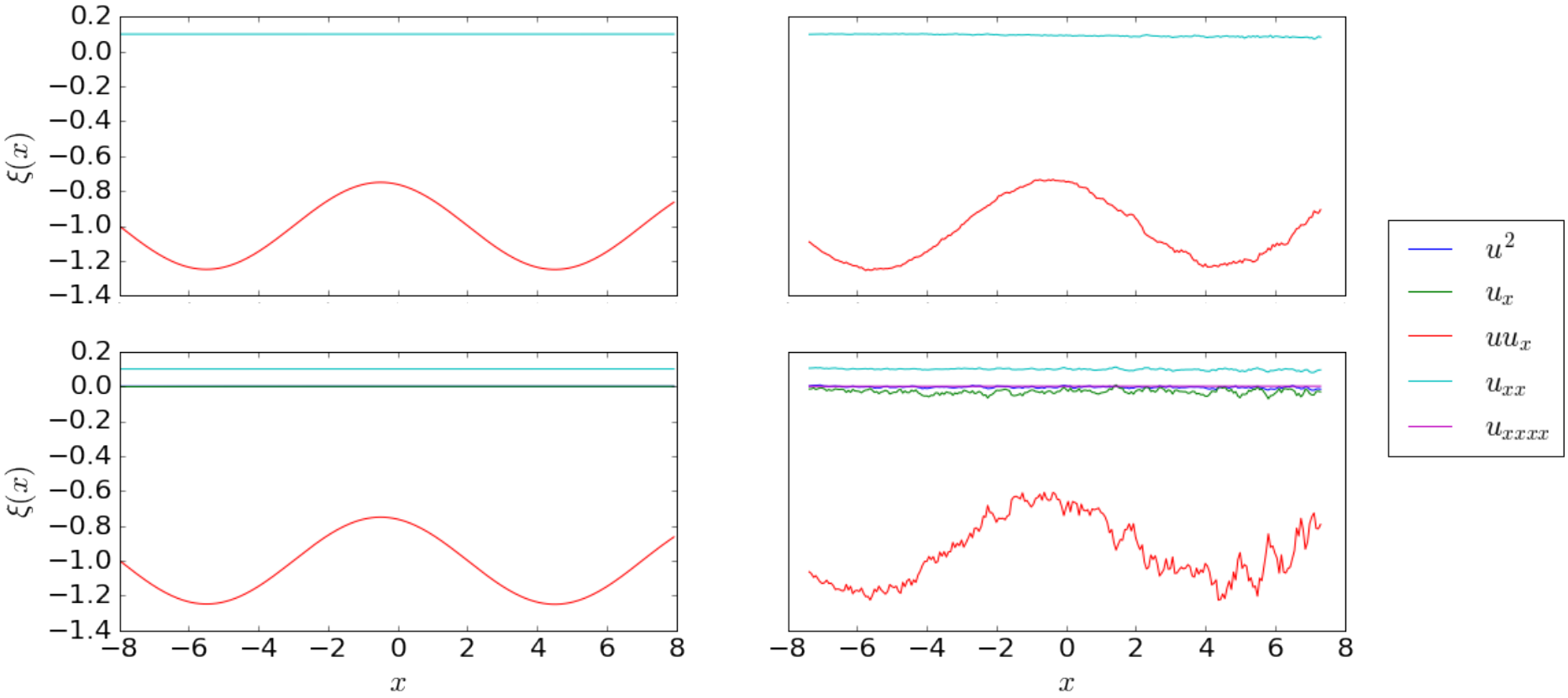
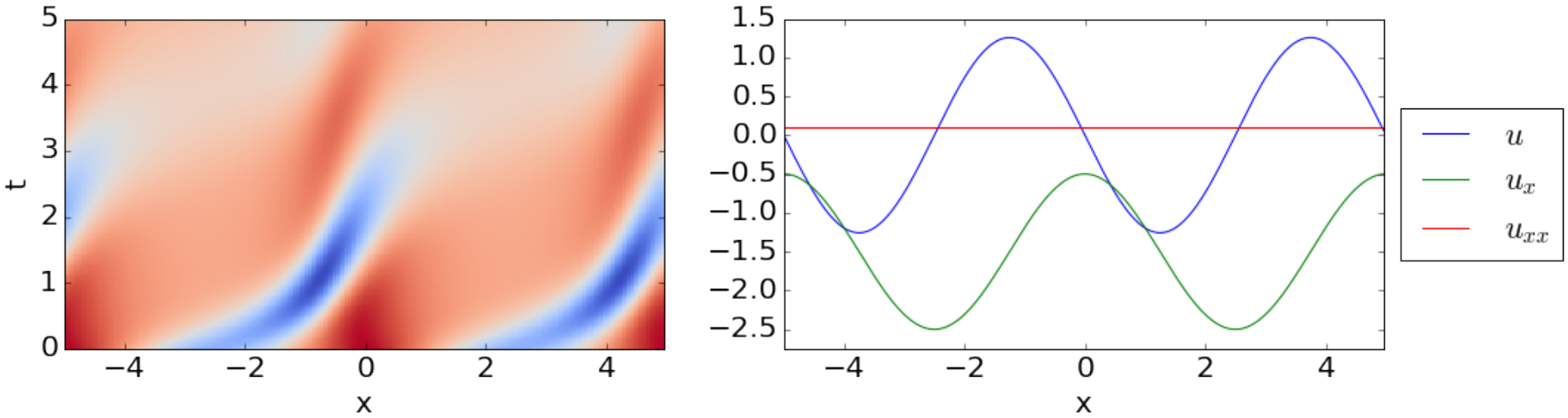


Figure: Time series discovered for the coefficients of the parametric Burgers' equation. Top row: SGTR method, which correctly identifies the two terms. Bottom row: group LASSO method which adds several additional (incorrect) terms to the model. The left panels are noise-free, while the right panels contain 1% noise.

Spatially Dependent Advection-Diffusion Equation



$$u_t = (c(x)u)_x + \epsilon u_{xx} = c(x)u_x + c'(x)u + \epsilon u_{xx}$$

Parameters

$x \in [-5, 5], t \in [0, 5], \epsilon = 0.1$, and $c(x) = -1.5 + \cos(2\pi x/L)$ using a spectral method with $n = 256$ and $m = 256$.

Library: powers of u up to cubic, multiplied by derivatives of u up to fourth order.

Spatially Dependent Advection-Diffusion Equation

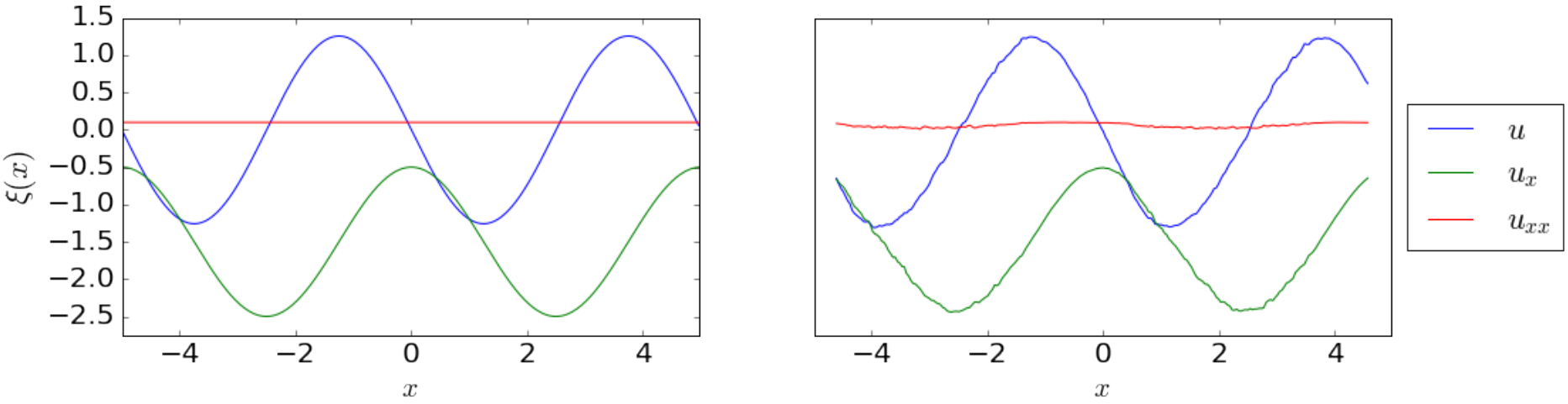


Figure: Spatial dependence of advection diffusion equation. Left: no noise. Right: 1% noise. Both SGTR and group LASSO correctly identified the active terms.

Conclusions and Future Directions

Conclusions

- We have presented DMD and its applications
  - Extrapolation of data
  - Reduction of the complexity of PDEs
  - Human mobility
- We have presented a method for identifying governing laws for physical systems which exhibit either spatially or temporally dependent behavior.

Outlook

- Solid mathematical background for DMD
- Identification of Stochastic Differential Equation (ongoing with Y. Sapurito)

References for this talk

- A. C. Balzotti, M. Briani, E. Cristian. *Understanding mass transfer directions via data-driven models with application to mobile phone data* , 2018
- A., J.N. Kutz. *Nonlinear model order reduction via dynamic mode deomposition*, 2017
- A., J.N. Kutz. *Randomized Model Order Reduction*, 2017
- C. Balzotti, A. Bragagnini, M. Briani, and E. Cristiani. *Understanding Human Mobility Flow from Aggregated Mobile Phone Data*, 2017
- S. L. Brunton, J. L. Proctor, J. N. Kutz. *Discovering governing equations from data by sparse identification of nonlinear dynamical systems,* 2016
- S.H. Rudy, A., S.L. Brunton, J.N. Kutz. *Data-driven identification of parametric partial differential equations*, 2018
- S.H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz. *Data-driven discovery of partial differential equations,* 2017
- P.J. Schmid, *Dynamic mode decomposition of numerical and experimental data,* 2010
- J.H. Tu, C.W. Rowley, D.M. Luchtenburg, S.L. Brunton, and J.N. Kutz. *On dynamic mode decomposition: Theory and applications,* 2014

THANK YOU FOR YOUR ATTENTION