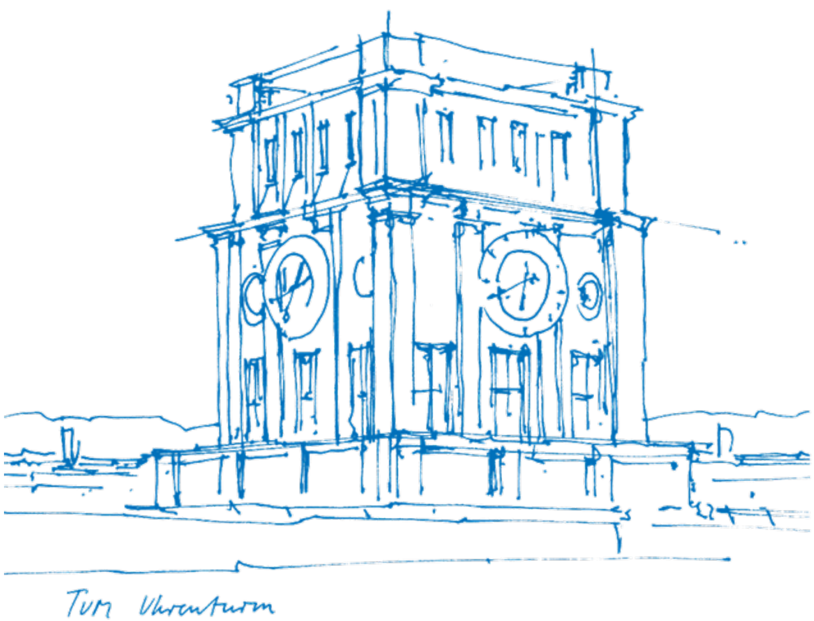
MLCMS, Lecture 5: Extracting dynamical systems from data

Felix Dietrich

2022-12-22—organizational issues



Organizational issues

Groups, Moodle, Reports

- Report due for exercise 4: 2022-12-22
- Report due for exercise 5: 2023-01-19
- Please send a one page summary of your planned final project until 2023-01-19
  - by email to me, `felix.dietrich@tum.de`,
  - and even better: a few days before, so I can give feedback and you can start right away.

Recap / Outlook

Lecture 1: Modeling crowd dynamics

- Modeling approaches, verification and validation

Lecture 2: Simulation software

- Introduction to the Vadere software, SIR models

Lecture 3: Representation of data

- Principal Component Analysis, Diffusion Maps, neural networks

Lecture 4: Dynamical systems and bifurcation theory

- Introduction to the theory and examples

Lecture 5: Extracting dynamical systems from data

- Function approximation, vector fields, time-delay embedding, (final projects)

Lecture 6: Future directions of machine learning

- Challenges in data science, master's thesis topics, final projects

Today: Machine Learning, take 2!

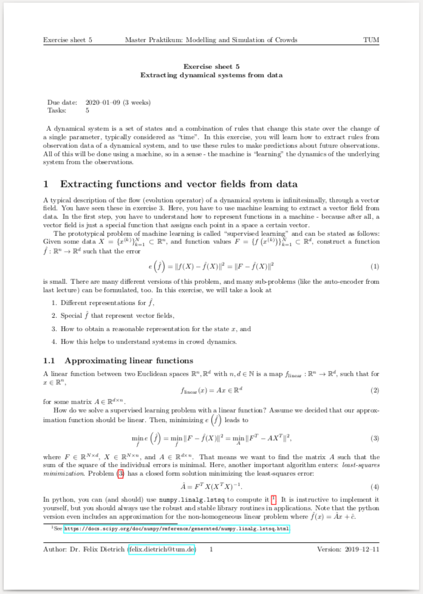
Extracting dynamical systems from data

1. Video 2: Approximating functions from data
2. Video 2: Vector fields from observations
3. Video 2: Time-delay embedding
4. Exercise 5
5. Final projects

Representation of data

Exercise 5

You can find the exercise sheet on Moodle



Exercise sheet 5 Master Praktikum: Modelling and Simulation of Crowds TUM

Exercise sheet 5
Extracting dynamical systems from data

Due date: 2020–01–09 (3 weeks)
Tasks: 5

A dynamical system is a set of states and a combination of rules that change this state over the change of a single parameter, typically considered as "time". In this exercise, you will learn how to extract rules from observation data of a dynamical system, and to use these rules to make predictions about future observations. All of this will be done using a machine, so in a sense - the machine is "learning" the dynamics of the underlying system from the observations.

1 Extracting functions and vector fields from data

A typical description of the flow (evolution operator) of a dynamical system is infinitesimally, through a vector field. You have seen these in exercise 3. Here, you have to use machine learning to extract a vector field from data. In the first step, you have to understand how to represent functions in a machine - because after all, a vector field is just a special function that assigns each point in a space a certain vector.

The prototypical problem of machine learning is called "supervised learning" and can be stated as follows: Given some data $X = \{x^{(k)}\}_{k=1}^N \subset \mathbb{R}^n$, and function values $F = \{f\left(x^{(k)}\right)\}_{k=1}^N \subset \mathbb{R}^d$, construct a function $\hat{f} : \mathbb{R}^n \to \mathbb{R}^d$ such that the error

$$e\left(\hat{f}\right) = \|f(X) - \hat{f}(X)\|^2 = \|F - \hat{f}(X)\|^2 \quad (1)$$

is small. There are many different versions of this problem, and many sub-problems (like the auto-encoder from last lecture) can be formulated, too. In this exercise, we will take a look at

1. Different representations for $\hat{f}$,
2. Special $\hat{f}$ that represent vector fields,
3. How to obtain a reasonable representation for the state x, and
4. How this helps to understand systems in crowd dynamics.

1.1 Approximating linear functions

A linear function between two Euclidean spaces $\mathbb{R}^n, \mathbb{R}^d$ with $n, d \in \mathbb{N}$ is a map $f_{\text{linear}} : \mathbb{R}^n \to \mathbb{R}^d$, such that for $x \in \mathbb{R}^n$,

$$f_{\text{linear}}(x) = Ax \in \mathbb{R}^d \quad (2)$$

for some matrix $A \in \mathbb{R}^{d \times n}$.

How do we solve a supervised learning problem with a linear function? Assume we decided that our approximation function should be linear. Then, minimizing $e\left(\hat{f}\right)$ leads to

$$\min_{\hat{f}} e\left(\hat{f}\right) = \min_{\hat{f}} \|F - \hat{f}(X)\|^2 = \min_{A} \|F^T - AX^T\|^2, \quad (3)$$

where $F \in \mathbb{R}^{N \times d}$, $X \in \mathbb{R}^{N \times n}$, and $A \in \mathbb{R}^{d \times n}$. That means we want to find the matrix A such that the sum of the square of the individual errors is minimal. Here, another important algorithm enters: *least-squares minimization*. Problem (3) has a closed form solution minimizing the least-squares error:

$$\hat{A} = F^T X (X^T X)^{-1}. \quad (4)$$

In python, you can (and should) use `numpy.linalg.lstsq` to compute it[1]. It is instructive to implement it yourself, but you should always use the robust and stable library routines in applications. Note that the python version even includes an approximation for the non-homogeneous linear problem where $\hat{f}(x) = \hat{A}x + \hat{c}$.

[1] See https://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.lstsq.html.

Author: Dr. Felix Dietrich (felix.dietrich@tum.de) 1 Version: 2019–12–11

Next lecture

Summary, outlook, final project

1. Lecture (but no exercise): summary and future topics of machine learning
2. Discussions of final projects (25% of your grade!)
   - 2.1 You have to decide on a final project for your group.
   - 2.2 Send me a one-page summary with five task descriptions for your project.
   - 2.3 On the day of the lecture (2023-01-19) we will briefly discuss the projects you chose.
   - 2.4 I will assign a presentation date for your group: 2023-01-26, 2023-02-02 , or 2023-02-09.
   - 2.5 You have to hand in the report until 2023-02-09 (three weeks time!).
   - 2.6 Grading for the final project is 1/3 code, 1/3 report, and 1/3 presentation.
   - 2.7 This year the presentations will be online, on Zoom.
   - 2.8 The following final project suggestions can be taken by multiple groups each, so no first-come-first-served. You still need to send me your project description page, though!

Final projects

(Example 1-page suggestion)

(from a previous semester)

Final projects

(Example 1) - Good visualization

Develop an efficient, robust and functional visualization for crowd trajectories.

Task1 Setting up the software environment: Unity? Unreal? Own code?

Task2 Two-dimensional visualization of individuals and obstacles

Task3 Additional visualization elements: trajectories, target zones

Task4 Basic user interaction: fast forward, jumping in time, zooming

Task5 Three-dimensional visualization

Final projects

(Example 2) - Learning dynamical systems from data: Neural networks

Implement Euler and Runge-Kutta templates in neural networks. This is a review and implementation of the papers [Rico-Martinez et al., 1994, Rico-Martínez and Kevrekidis, 1995].

Task1 Summary of the paper contents

Task2 Setting up the neural network for Euler's method

Task3 Setting up the neural network for the Runge-Kutta method

Task4 Testing the networks in a two-dimensional dynamical system example

Task5 Constructing bifurcation diagrams using the extracted dynamics

Final projects

(Example 3) - Adding a module to Vadere

Work with the Vadere software and add an interesting module (velocity processor, simple bifurcation analysis, 3D visualization).

Task1 Description of the module structure: requirements, software, etc.

Task2 Stand-alone implementation for testing

Task3 Integration into Vadere

Task4 Testing the module on a crowd dynamics example

Task5 Unit tests

Final projects

(Example 4) - Learning dynamical systems from data: Koopman

Study the Koopman operator framework [Budišić et al., 2012, Williams et al., 2015b, Williams et al., 2015a, Li et al., 2017, Mauroy and Goncalves, 2017, Dietrich et al., 2020] and implement/understand the Extended Dynamic Mode Decomposition in [Williams et al., 2015b] (the `datafold` software already implements EDMD).

Task1 Description of the Koopman operator

Task2 First, own implementation of the EDMD algorithm

Task3 Tests on an example in the paper by Williams et al.

Task4 Tests on a simple example in crowd dynamics (I will give you one)

Task5 Discussion of the results

Final projects

(Example 5) - Learn and visualize representations for large data sets

This is a review and implementation of the paper [McQueen et al., 2016]. Choose if you want to use “datafold”, or test neural network (auto-encoder) performance against theirs.

Task1 Description of the data sets

Task2 Using existing implementation of 2-3 fast representation algorithms

Task3 Tests on a large dataset you make up yourself

Task4 Tests on one of the large datasets in the paper

Task5 Comparison to their results

Final projects

(Example 6) - Redo a bifurcation scenario from Prof. Starkes papers

You may have already read one of the bifurcation analysis papers by the group of Prof. Starke [Starke et al., 2014, Marschler et al., 2014, Marschler, 2014]. Decide on one and try to reproduce their results.

Task1 Description of the example, discussion of challenges and your choice of method

Task2 Implementation of the bifurcation analysis

Task3 Tests on a simple example

Task4 Tests on one of the examples in the paper

Task5 Comparison to their results

Final projects

(Example 7) - Discuss a neural network paper for crowd dynamics

Do a literature review of [Tordeux et al., 2019]. Decide on one example and try to reproduce their results. Careful: they chose a very special neural architecture (very few neurons), so the discussion must be about that, too!

Task1 Description of the example, discussion of challenges and your choice of method

Task2 Implementation of the neural network

Task3 Tests on a simple example

Task4 Comparison to their results

Task5 Discussion of the approach and architecture.

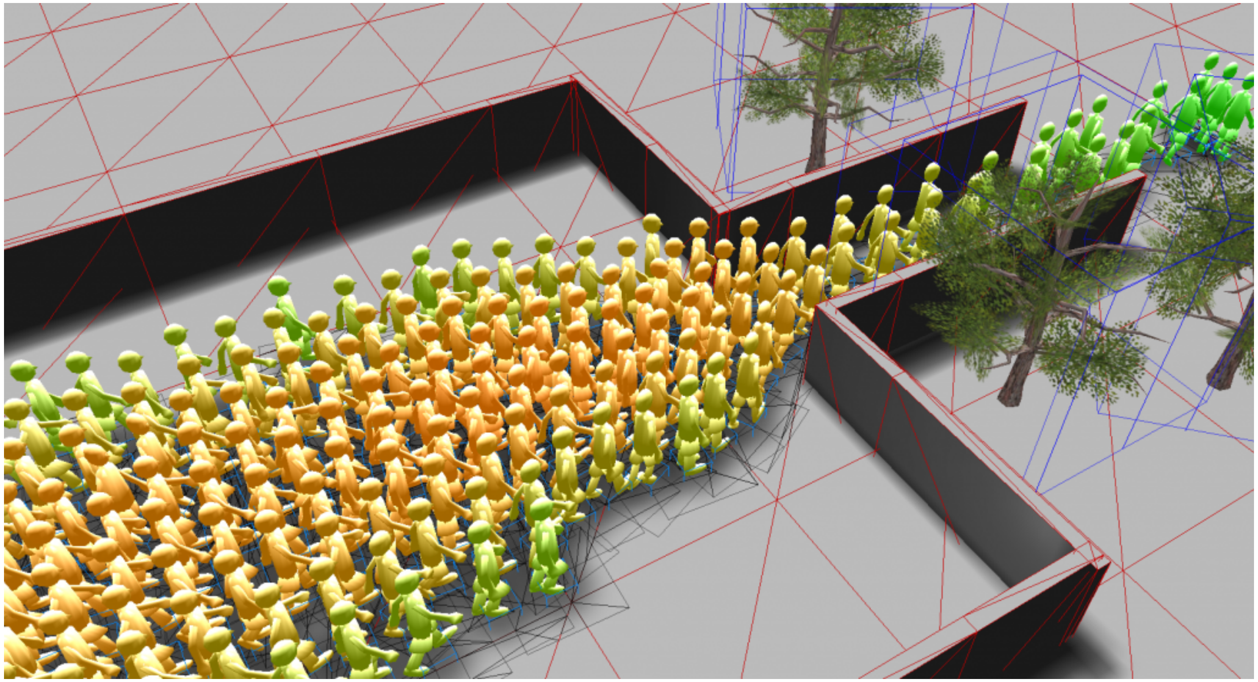
Final projects

(Example 8) - Pedestrian trajectory prediction methods: deep learning and knowledge-based approaches

Discuss the two directions from the review of [Korbmacher and Tordeux, 2022].

- Task1 Introduction of the two approaches separately.
- Task2 Discussion of the benefits.
- Task3 Discussion of the drawbacks and challenges.
- Task4 Comparison of the approaches.
- Task5 Discussion of future work and other approaches, separate from the review paper [Korbmacher and Tordeux, 2022].

Questions?



Homework 1: finish fourth exercise & upload report until 2022-12-22.
Homework 2: finish fifth exercise & upload report until 2023-01-19.
Final project: Send me a list of five tasks with a short description until 2023-01-19.
For questions / appointments: please ask via email, `felix.dietrich@tum.de`.

Literature I

Budišić, M., Mohr, R., and Mezić, I. (2012).
Applied Koopmanism.
Chaos: An Interdisciplinary Journal of Nonlinear Science, 22:047510.

Dietrich, F., Thiem, T. N., and Kevrekidis, I. G. (2020).
On the Koopman Operator of Algorithms.
SIAM Journal on Applied Dynamical Systems, 19(2):860–885.

Korbmacher, R. and Tordeux, A. (2022).
Review of Pedestrian Trajectory Prediction Methods: Comparing Deep Learning and Knowledge-based Approaches.

Li, W., Li, Y., Yu, P., Gong, J., Shen, S., Huang, L., and Liang, J. (2017).
Modeling, simulation and analysis of the evacuation process on stairs in a multi-floor classroom building of a primary school.
Physica A: Statistical Mechanics and its Applications, 469:157–172.

Marschler, C. (2014).
Coarse Analysis of Microscopic Models Using Equation-Free Methods.
PhD thesis, Technical University of Denmark.

Marschler, C., Sieber, J., Berkemer, R., Kawamoto, A., and Starke, J. (2014).
Implicit Methods for Equation-Free Analysis: Convergence Results and Analysis of Emergent Waves in Microscopic Traffic Models.
SIAM Journal on Applied Dynamical Systems, 13(3):1202–1238.

Mauroy, A. and Goncalves, J. (2017).
Koopman-Based Lifting Techniques for Nonlinear Systems Identification.
IEEE Transactions on Automatic Control.

McQueen, J., Meilă, M., VanderPlas, J., and Zhang, Z. (2016).
Megaman: Scalable Manifold Learning in Python.
Journal of Machine Learning Research, 17(148):1–5.

Rico-Martinez, R., Anderson, J. S., and Kevrekidis, I. G. (1994).
Continuous-time nonlinear signal processing: A neural network based approach for gray box identification.
In *Proceedings of IEEE Workshop on Neural Networks for Signal Processing*. IEEE.

Rico-Martínez, R. and Kevrekidis, I. G. (1995).
Nonlinear system identification using neural networks: Dynamics and instabilities.
In *Neural Networks for Chemical Engineers*, chapter 16, pages 409–442. Elsevier Science.

Starke, J., Thomsen, K. B., Soerensen, A., Marschler, C., Schilder, F., Dederichs, A., and Hjorth, P. (2014).
Nonlinear Effects in Examples of Crowd Evacuation Scenarios.
In *IEEE 17th International Conference on Intelligent Transportation Systems (ITSC), October 8-11, 2014,Qingdao, China*, Qingdao, China.

Tordeux, A., Chraibi, M., Seyfried, A., and Schadschneider, A. (2019).
Prediction of pedestrian dynamics in complex architectures with artificial neural networks.
Intelligent Transportation Systems, 24(6):556–568.

Literature II

Williams, M. O., Kevrekidis, I. G., and Rowley, C. W. (2015a).
A Data-Driven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition.
Journal of Nonlinear Science, 25(6):1307–1346.

Williams, M. O., Rowley, C. W., Mezić, I., and Kevrekidis, I. G. (2015b).
Data fusion via intrinsic dynamic variables: An application of data-driven Koopman spectral analysis.
EPL (Europhysics Letters), 109(4).