ἀρχὰς εἶναι τῶν ὅλων ἀτόμους καὶ κενόν,

τὰ δ'ἀλλα πάντα νενομίσθαι

一切事物的基本元素是原子和虚空，
其余的都是人的看法。

Los elementos de todo son átomos y vacío,
todo lo demás es mera opinión.

عناصر همه چیز اتمها و خلأ هستند
، همه چیز دیگر تنها نظر است

היסודות של הכול הם אטומים וריק
, כל השאר הוא רק דעה.

Элементы всего — это атомы и пустота,
все остальное — лишь мнение

عناصر كل شيء هي الذرات والفراغ
، وكل ما عدا ذلك هو مجرد رأي.

Die Elemente von allem sind Atome und Leere,
alles andere ist nur Meinung.

모든 것의 요소는 원자와 공허이며,
나머지는 단지 의견일 뿐입니다.

Les éléments de tout sont les atomes et le vide,
tout le reste n'est que opinion.

Os elementos de tudo são átomos e vazio,
todo o resto é mera opinião.

Vipengele vya kila kitu ni atomu na utupu,
yote mengine ni maoni tu

সবকিছুর উপাদান হল পরমাণু এবং শূন্য,
বাকি সবই শুধুমাত্র মতামত।

सब कुछ के तत्व अणु और शून्य हैं,
बाकी सब केवल राय है।

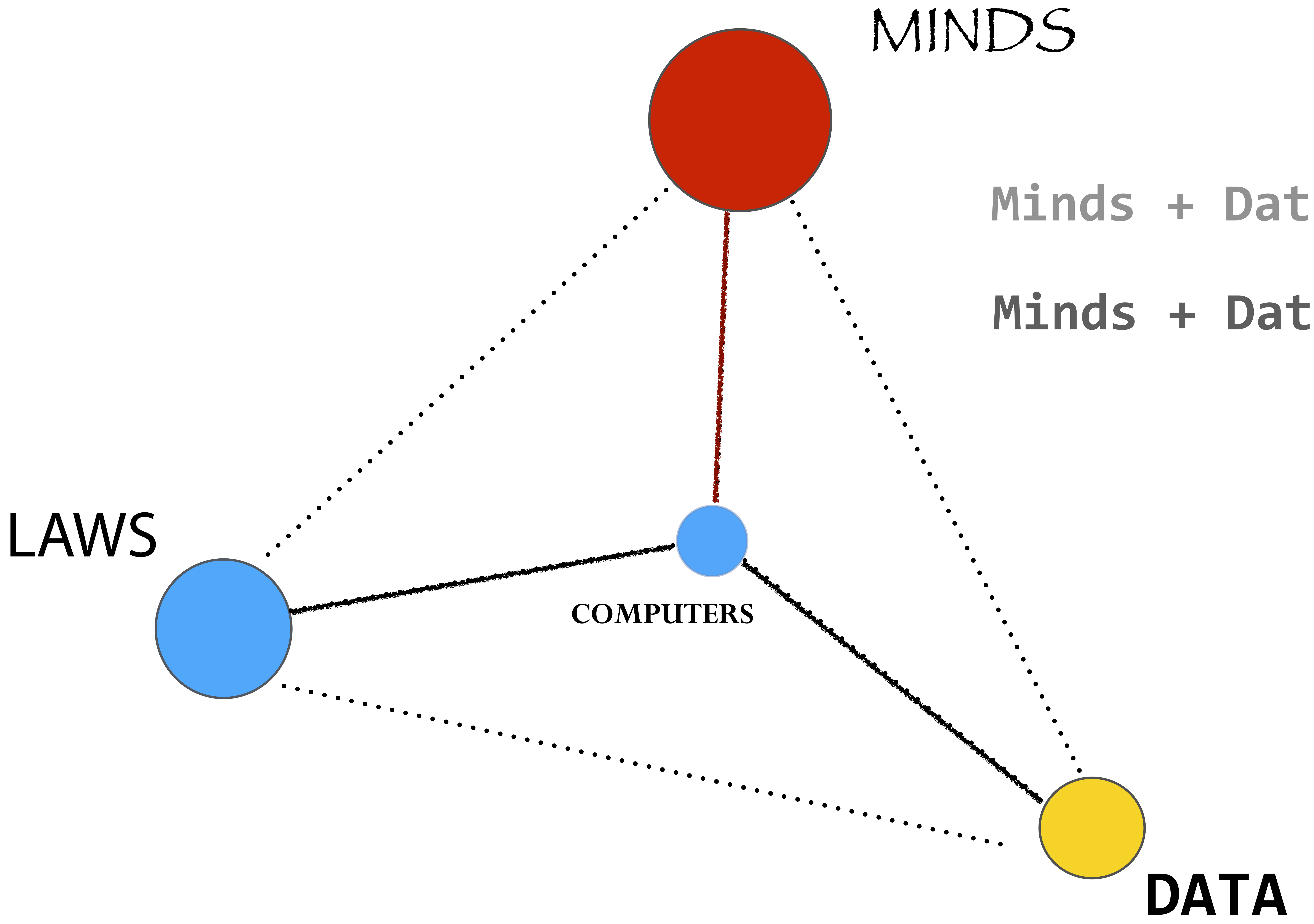
Các yếu tố của mọi thứ là nguyên tử và khoảng không,
tất cả những thứ khác chỉ là ý kiến.

すべての要素は原子と空虚であり、
その他のすべては単なる意見です。

Her şeyin unsurları atomlar ve boşluktur,
geri kalan her şey sadece fikirdir.

The elements of everything are atoms and void,

everything else is mere opinion



Minds + Data + Machines = Computing

Minds + Data + Machines = AI

Minds + Data + Machines = ALLOYS

Petros Koumoutsakos

HARVARD
School of Engineering and Applied Sciences

LECTURES

- **LECTURE 1:**
  - What is Artificial Intelligence? How it relates to Computing ?
  - How are these relevant to modeling, understanding, optimizing complex systems?
- **LECTURE 2:**
  - (Machine) Learning of Effective Dynamics of Complex Systems.
  - Reinforcement Learning for Controlling Complex Systems.
- **LECTURE 3**
  - Learning to solve PDEs without Machine Learning
  - Learning to solve PDEs with Reinforcement Learning Machine
- **LECTURE 4**
  - Uncertainty Quantification & Stochastic Optimization
  - Optimal Sensor Placement

LECTURE 1

AI and COMPUTING


-2500 YEARS
RAFAEL

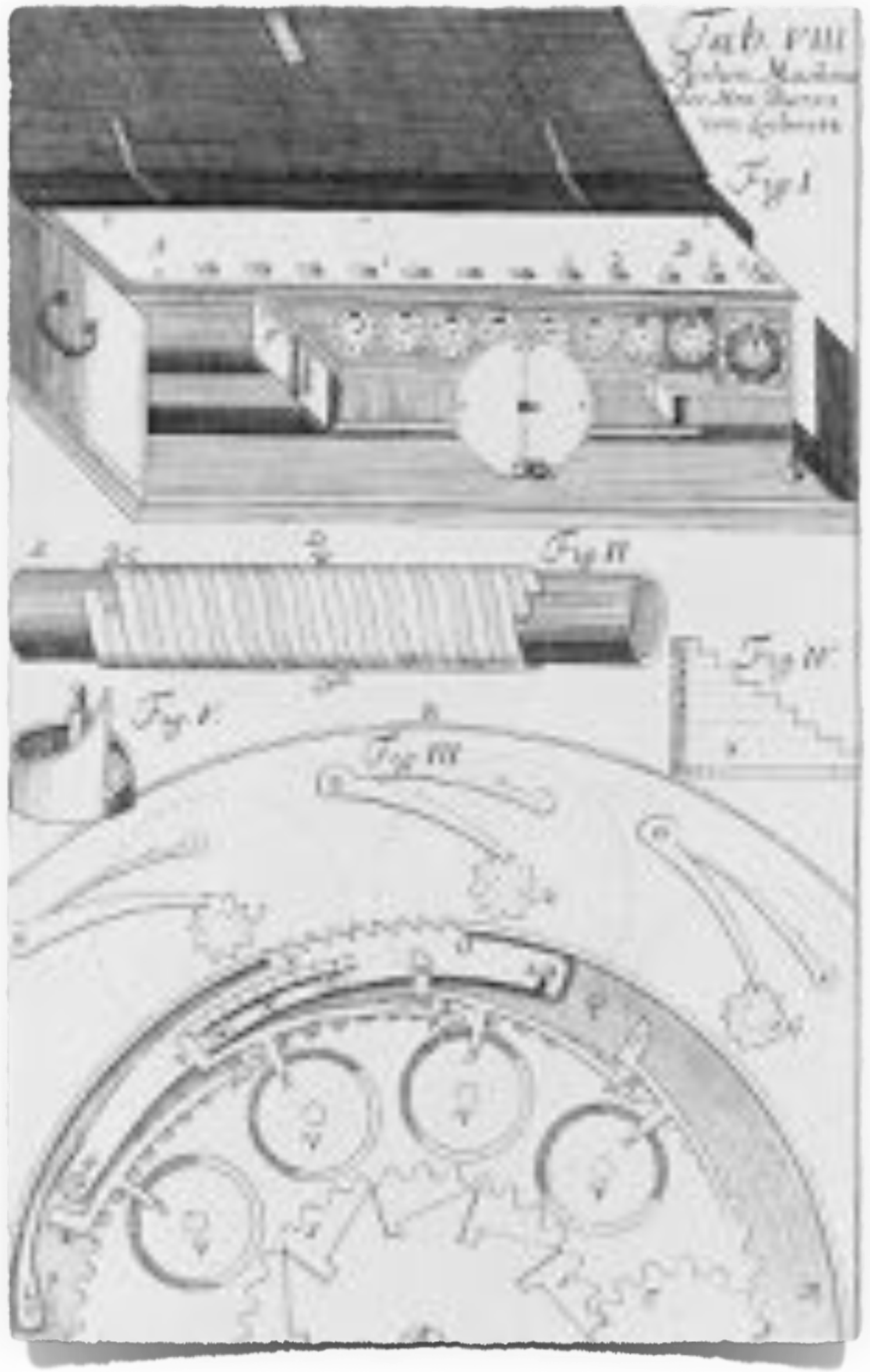
THE PURSUIT OF TRUTH

PLATO: *The Allegory of the Cave*

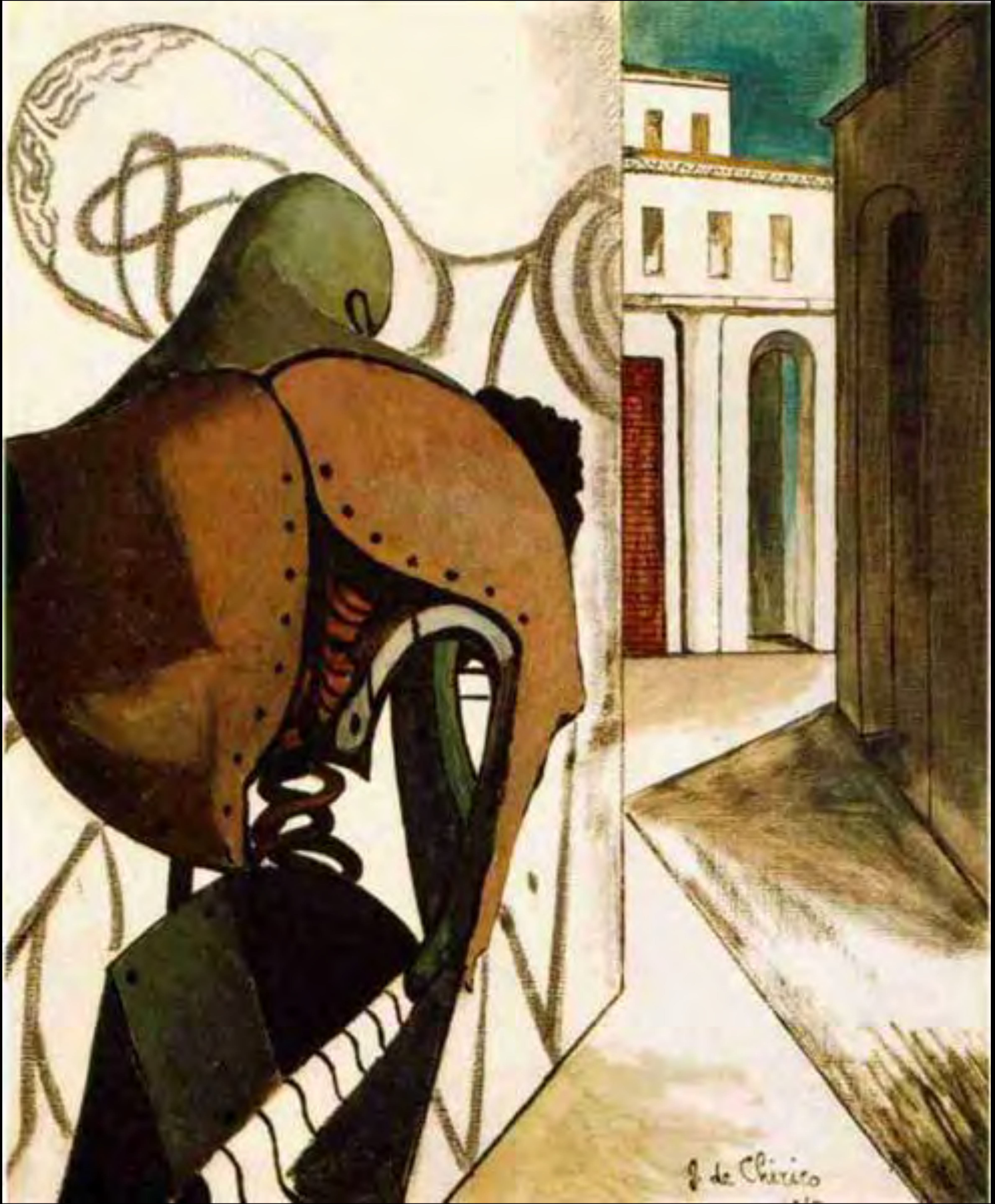
CREDIT: TED Ed

If **controversies** were to arise,
there would be no more need of disputation between
two philosophers than between **two accountants.**
For it would suffice to take their pencils in their hands,
and say to each other:

Calculemus—Let us calculate.



Euclid Descartes Russel Llull Hilbert Boole **Leibniz** Frege Newton Laplace Wittgenstein Turing Shannon

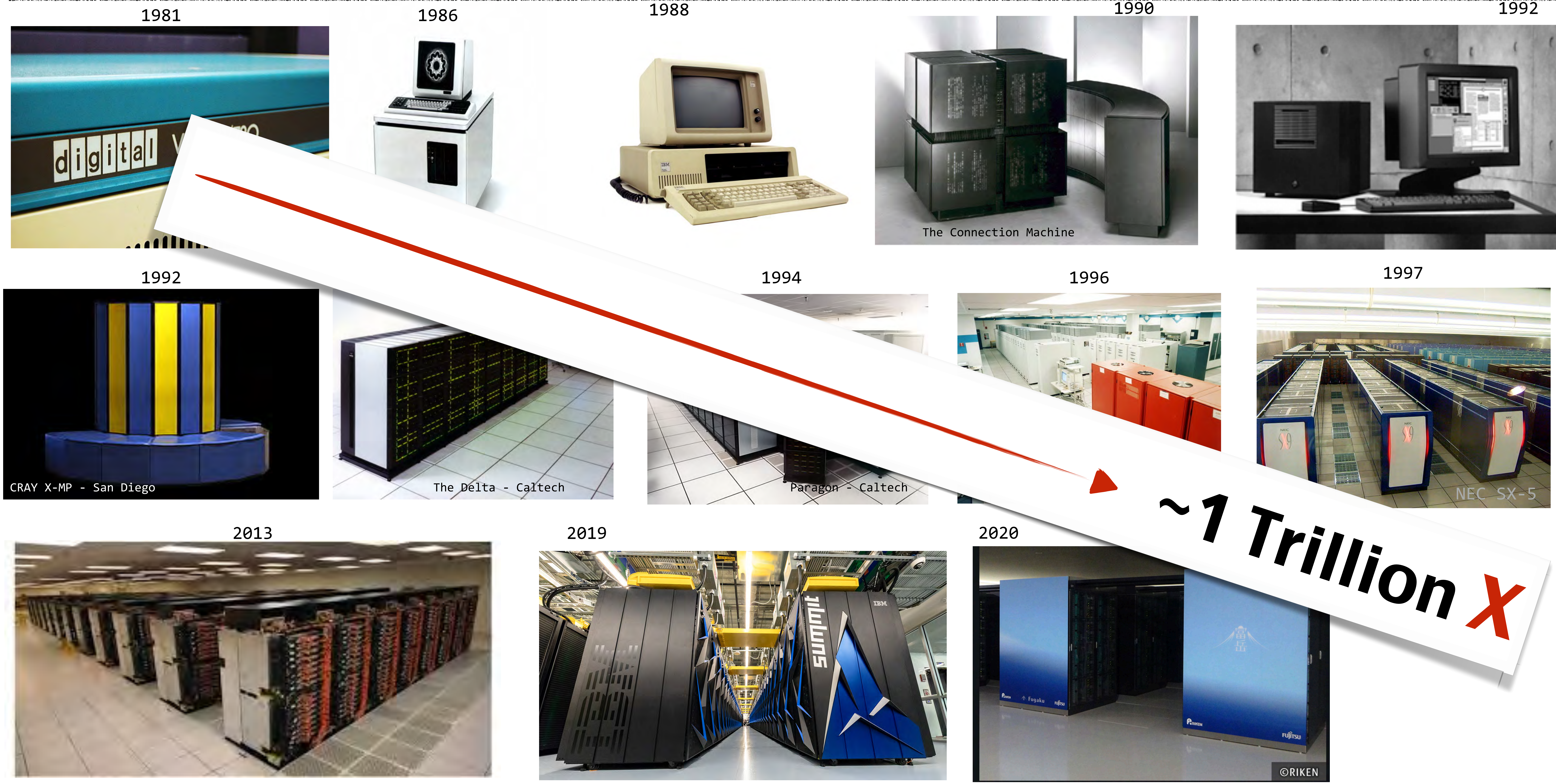

G. de Chirico


William Skyvington
Machina sapiens
Essai sur l'intelligence artificielle
GOURMELIN
Seuil

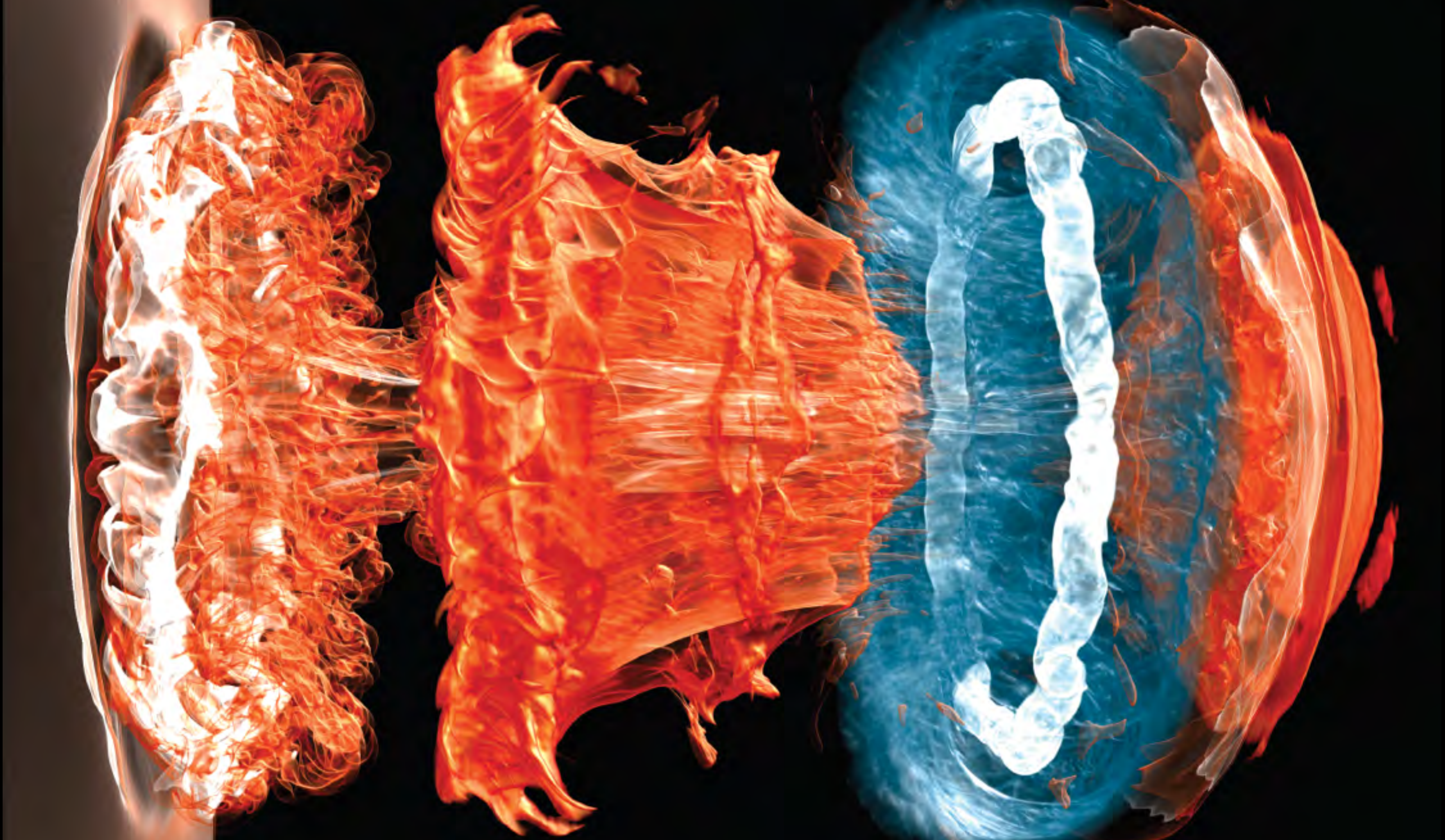
COMPUTING: The beginning…

1961

COMPUTERS



SOURCES: http://www.computerhistory.org/timeline/computers/



COMPUTERS : A Disruptive Technology

Deep Blue beat Kasparov

Posted by: Marco van der Spek Date: Oct 2, 2012

Category: Articles

http://www.testnewsonline.com/2012/10/02/deep-blue-beat-kasparov-because-of-bug/

MindGoogle) winning Go against Lee Sedol, one of the world's top go players.March 11, 2016

SCIENCE FILE - Los Angeles Times
9 March 2017

No need for a poker face - Software program DeepStack beats the pros at Texas Hold 'Em

ARTICLE

doi:10.1038/nature16961

Mastering the game of Go with deep neural networks and tree search

David Silver[1]*, Aja Huang[1]*, Chris J. Maddison[1], Arthur Guez[1], Laurent Sifre[1], George van den Driessche[1], Julian Schrittwieser[1], Ioannis Antonoglou[1], Veda Panneershelvam[1], Marc Lanctot[1], Sander Dieleman[1], Dominik Grewe[1], John Nham[2], Nal Kalchbrenner[1], Ilya Sutskever[2], Timothy Lillicrap[1], Madeleine Leach[1], Koray Kavukcuoglu[1], Thore Graepel[1] & Demis Hassabis[1]

DATA

2017:

5B Devices

1B people

2020:

32B Devices

2B people

http://electronicsofthings.com/research-articles/internet-of-things-and-big-data-predict-and-change-the-future/

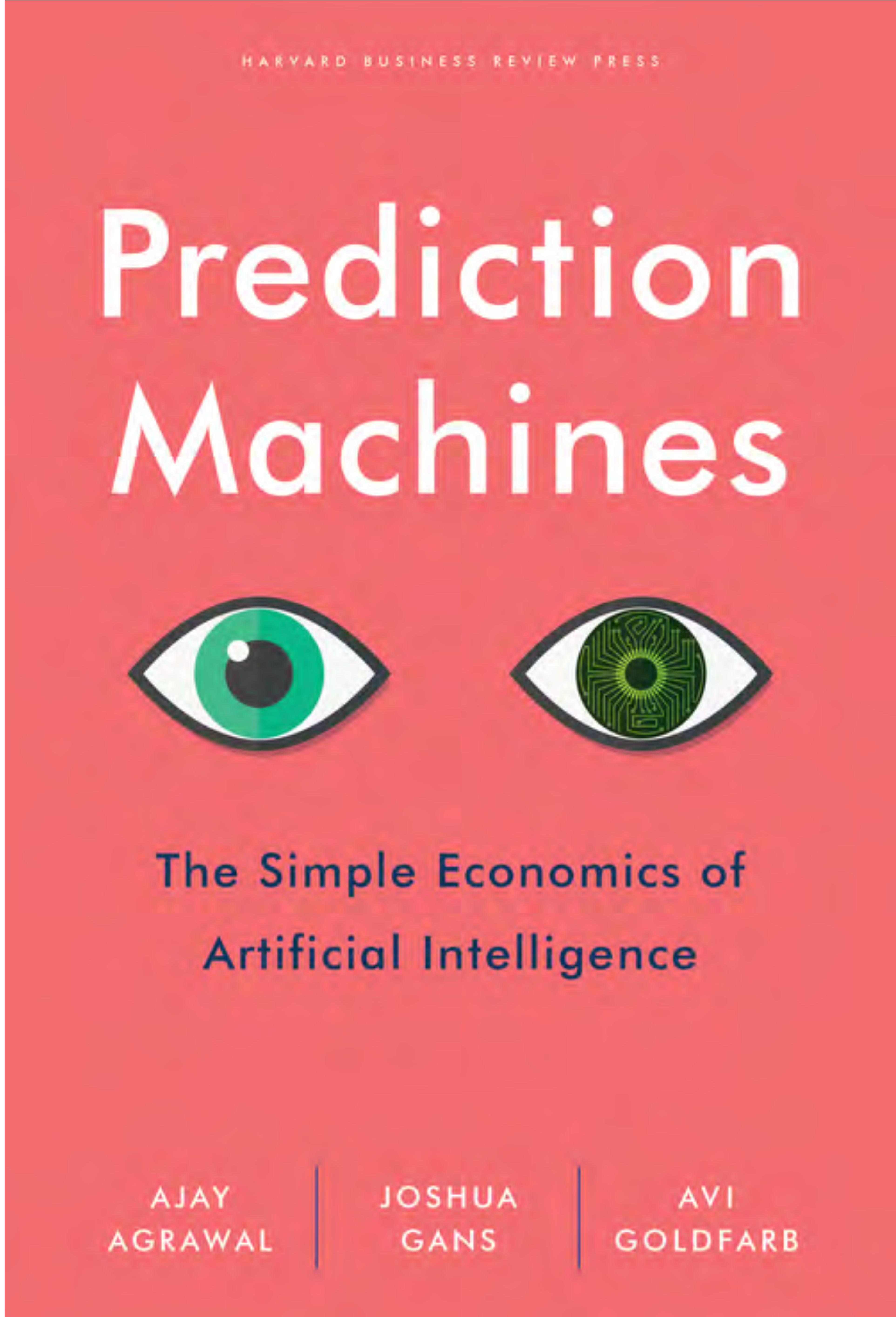
...computers have made arithmetic cheap.

Solving complex equations is done more easily and in less time ..

- **What will AI technology make cheap ? Prediction.**

- **Prediction is central to decision-making under uncertainty**

- **Better prediction under uncertainty -> new opportunities for all companies**



Whereas others see transformational new innovation, we see a simple fall in price.


Prediction vs Intelligence

What is Intelligence ?


John McCarthy

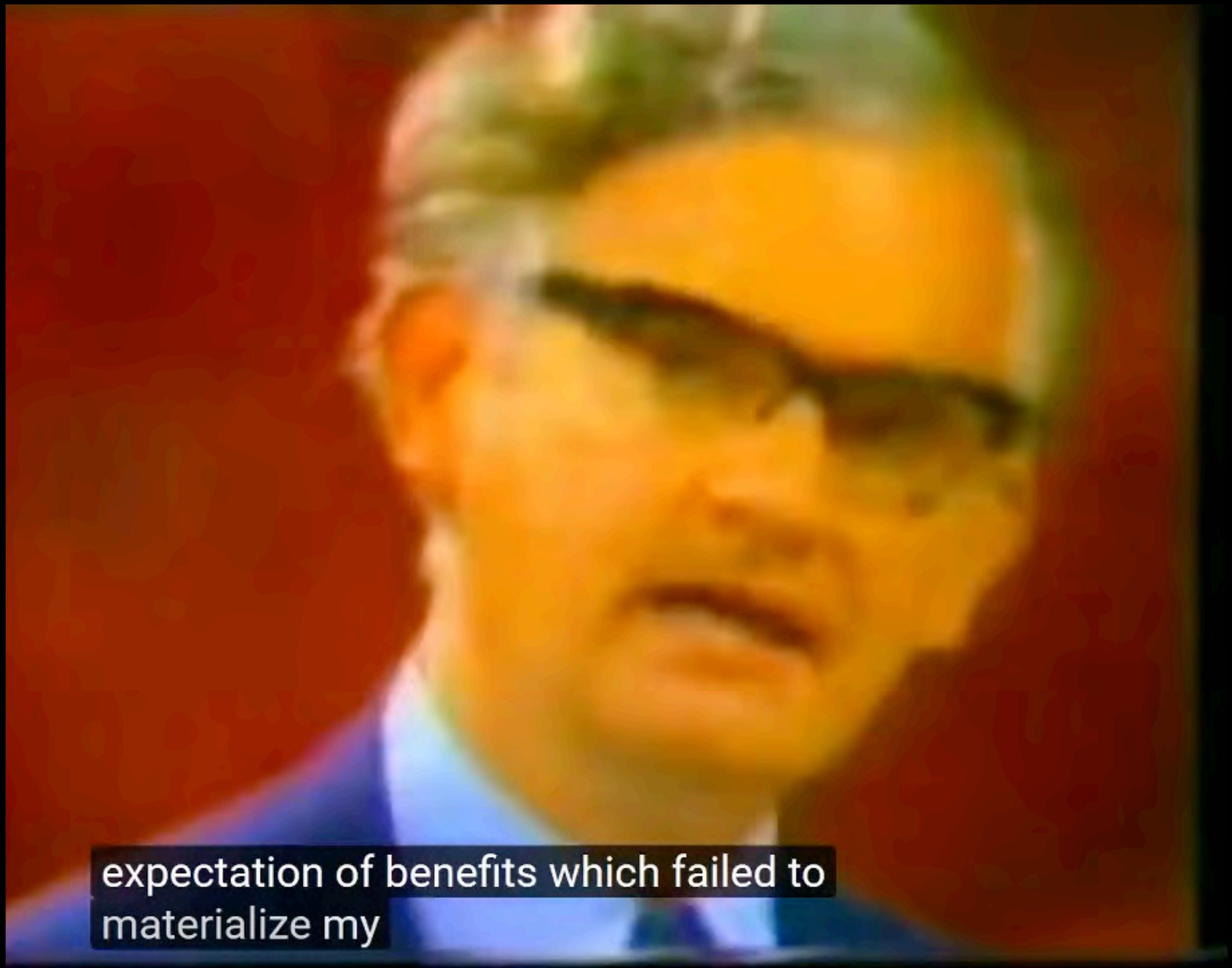
Intelligence is the computational part of the ability to achieve goals in the world.

A system having a goal or not, is not a property of the system itself. It is in the **relationship between the system and an observer.**

The system is most usefully understood/predicted/controlled in terms **of its outcomes rather than its mechanisms.**

http://jmc.stanford.edu/artificial-intelligence/what-is-ai/index.html

AI and Fluid Mechanics - The Lighthill Report (1973)



Lighthill's position does not come as a surprise. **He was, after all, a researcher in fluid dynamics and aeroacoustics, where it is easy to be misled by complicated differential equations involving 'continuous' variables and where nonexistent solutions arise so often.**

http://www.mathrix.org

Lighthill's main argument was that because one had to specify the rules in a computer to tell the robot how to behave in every possible scenario, every attempt to come up with a general purpose robot would quickly turn out to be an intractable problem, with a combinatorial explosion of possible solutions.

WHAT is Computing ?

*Computing is a domain of knowledge dealing with the study of Information processing, both **what** can be computed and **how** to compute it.* - Joseph Sifakis

QUESTION

MODEL

ALGORITHM

SOFTWARE

ARCHITECTURE

SIMULATION

DATA

ANALYSIS

DECISION

$$\frac{dR}{dt} = \alpha \cdot R \cdot G - \beta \cdot R \cdot F$$

RABBITS

Eat GRASS and give Birth

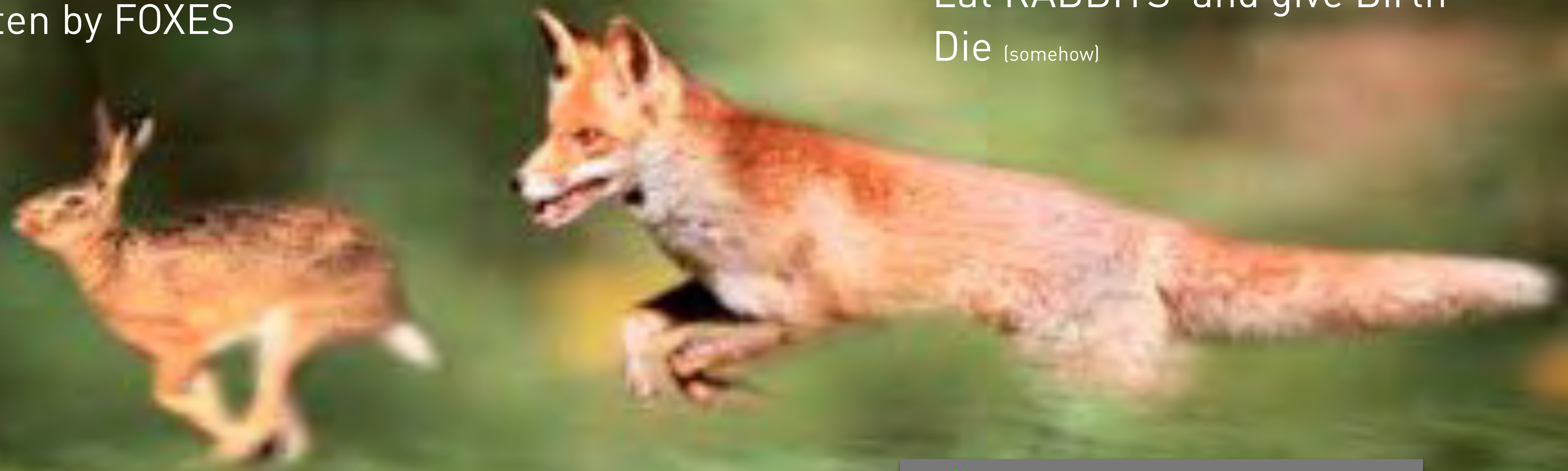
Die Eaten by FOXES

$$\frac{dF}{dt} = \beta \cdot R \cdot F - \gamma \cdot F$$

FOXES

Eat RABBITS and give Birth

Die (somehow)



GRASS

Grows by Rain

Dies Eaten by RABBITS and Pollution

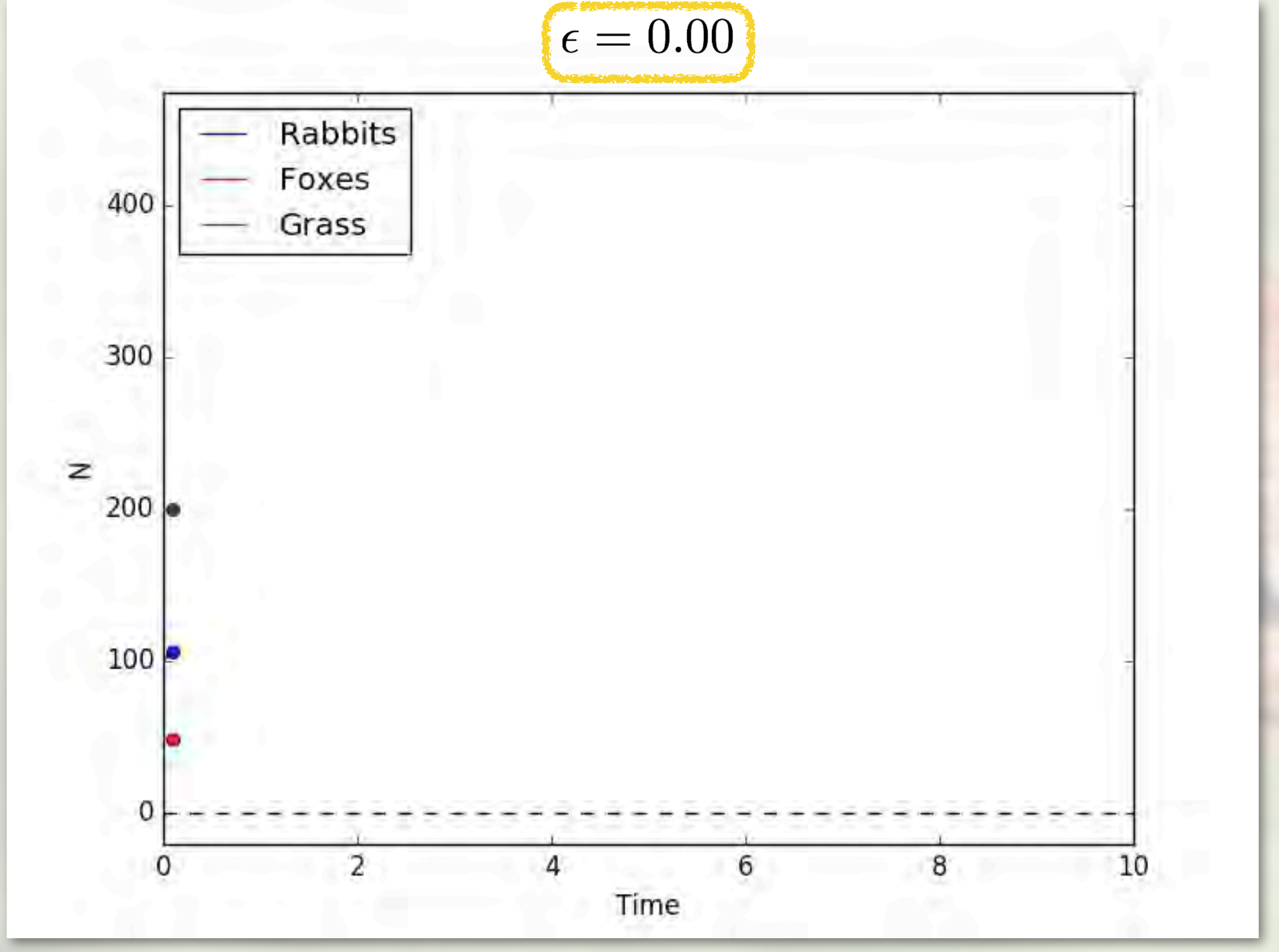
$$\frac{dG}{dt} = -\alpha \cdot R \cdot G + \delta \cdot G - \epsilon \cdot G$$

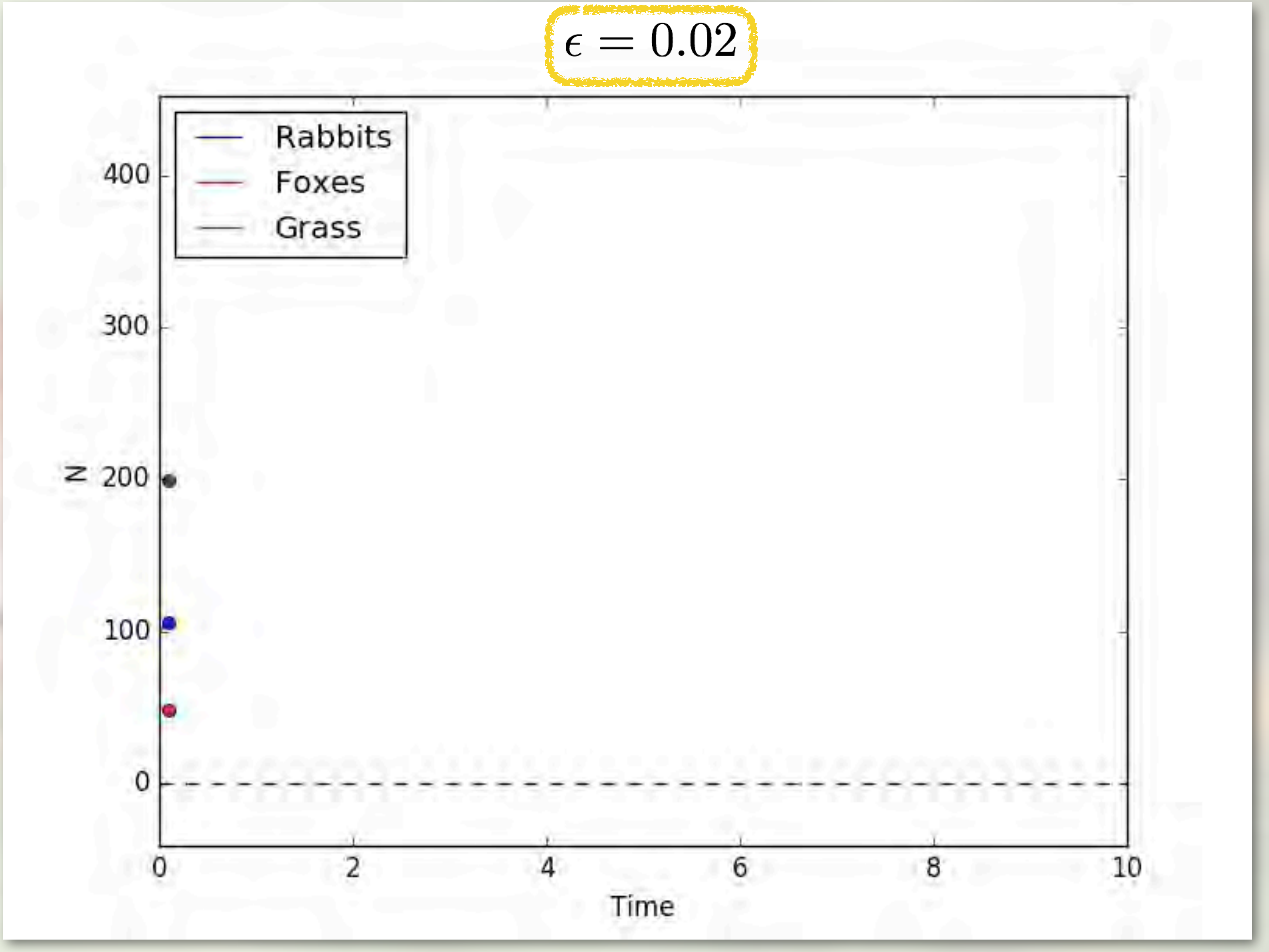
```python
def set_data(self):
    dt = 0.0001
    tmax = 100.
    self.times = np.arange(0., tmax, dt)
    states = odeint(self.dxdt, [X_0, Y_0, Z_0], self.times)

    #skip some data:
    self.x_ary, self.y_ary, self.z_ary = states[0::1000,0], states[0::1000,1],
        states[0::1000,2]
    self.times, self.N = self.times[0::1000], len(self.x_ary)
    self.ybot = min(np.amin(self.y_ary), np.amin(self.x_ary))
    self.ybot = min(np.amin(self.z_ary), self.ybot)
    self.ytop = max(np.amax(self.y_ary), np.amax(self.x_ary))
    self.ytop = max(np.amax(self.z_ary), self.ytop)
    self.ybot = self.ybot - 0.1*(self.ytop-self.ybot)
    self.ytop = self.ytop + 0.1*(self.ytop-self.ybot)

def data_stream(self):
    for i in range(self.N):
        yield self.x_ary[i], self.y_ary[i], self.z_ary[i], self.times[i], i

def show(self, bSaveMovie=False):
    if bSaveMovie:
        self.ani.save('lv.mov', fps=30, extra_args=['-vcodec', 'libx264', '-pix_fmt',
            'yuv420p'])
    else:
        plt.show()
```

$$\frac{dG}{dt} = -\alpha \cdot R \cdot G + \delta \cdot G - \epsilon \cdot G$$

$$\frac{dR}{dt} = \alpha \cdot R \cdot G - \beta \cdot R \cdot F$$

$$\frac{dF}{dt} = \beta \cdot R \cdot F - \gamma \cdot F$$





Predator Prey near extinction

- Are the equations valid for small numbers of foxes and rabbits ?

WHY
Compute
?

COMPLEMENT THEORY/EXPERIMENTS

TEST HYPOTHESES

KNOWLEDGE

PROCESS/ANALYZE DATA

PREDICTION

OPTIMIZE/DESIGN

DECIDE

KNOWLEDGE

How

Classical

Επιστημη

Principles/Laws

Science

C O M P U T I N G

What

Empirical

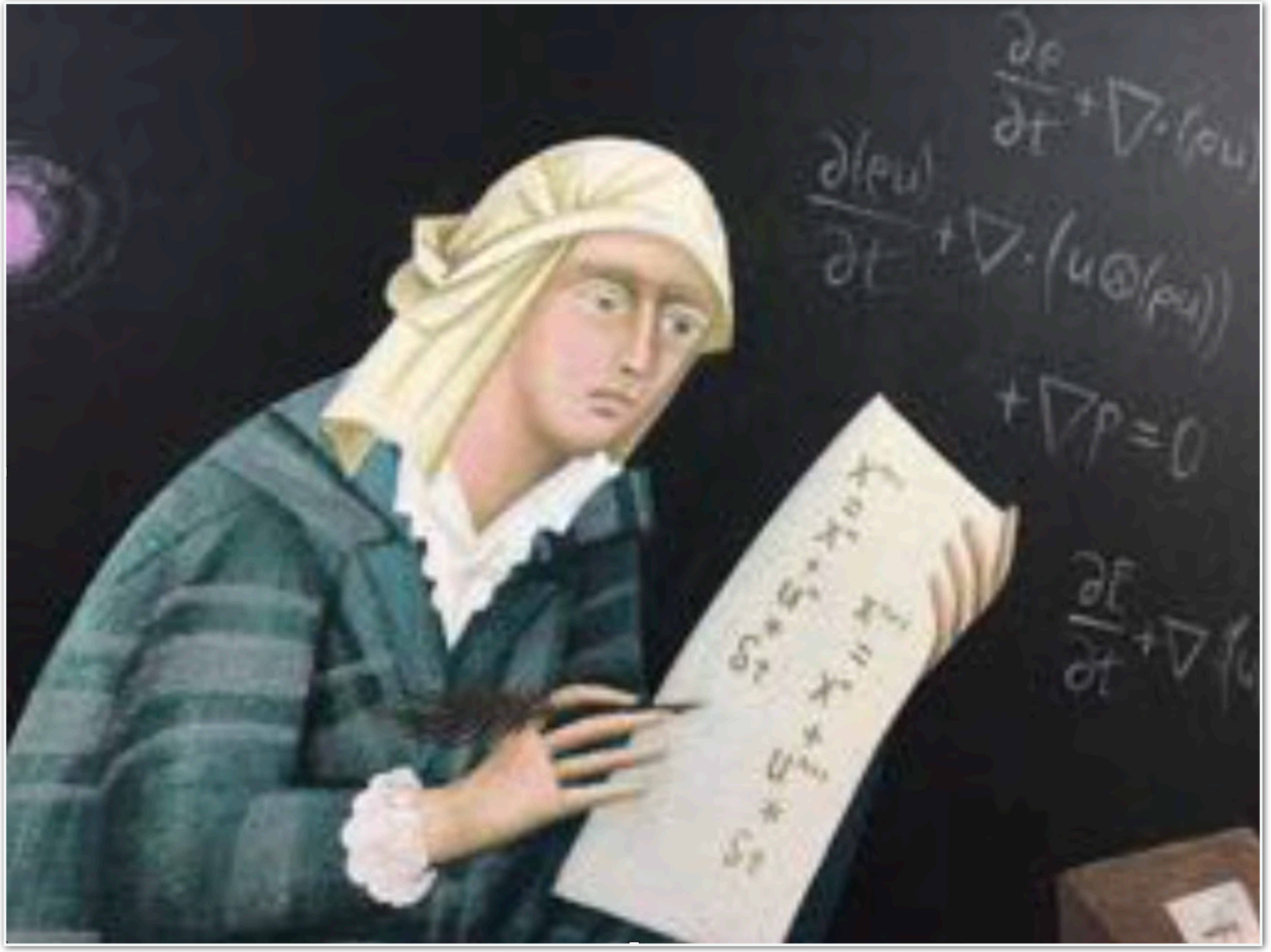
Τεχνη

Heuristics/Structures

Engineering

CLASSICAL KNOWLEDGE

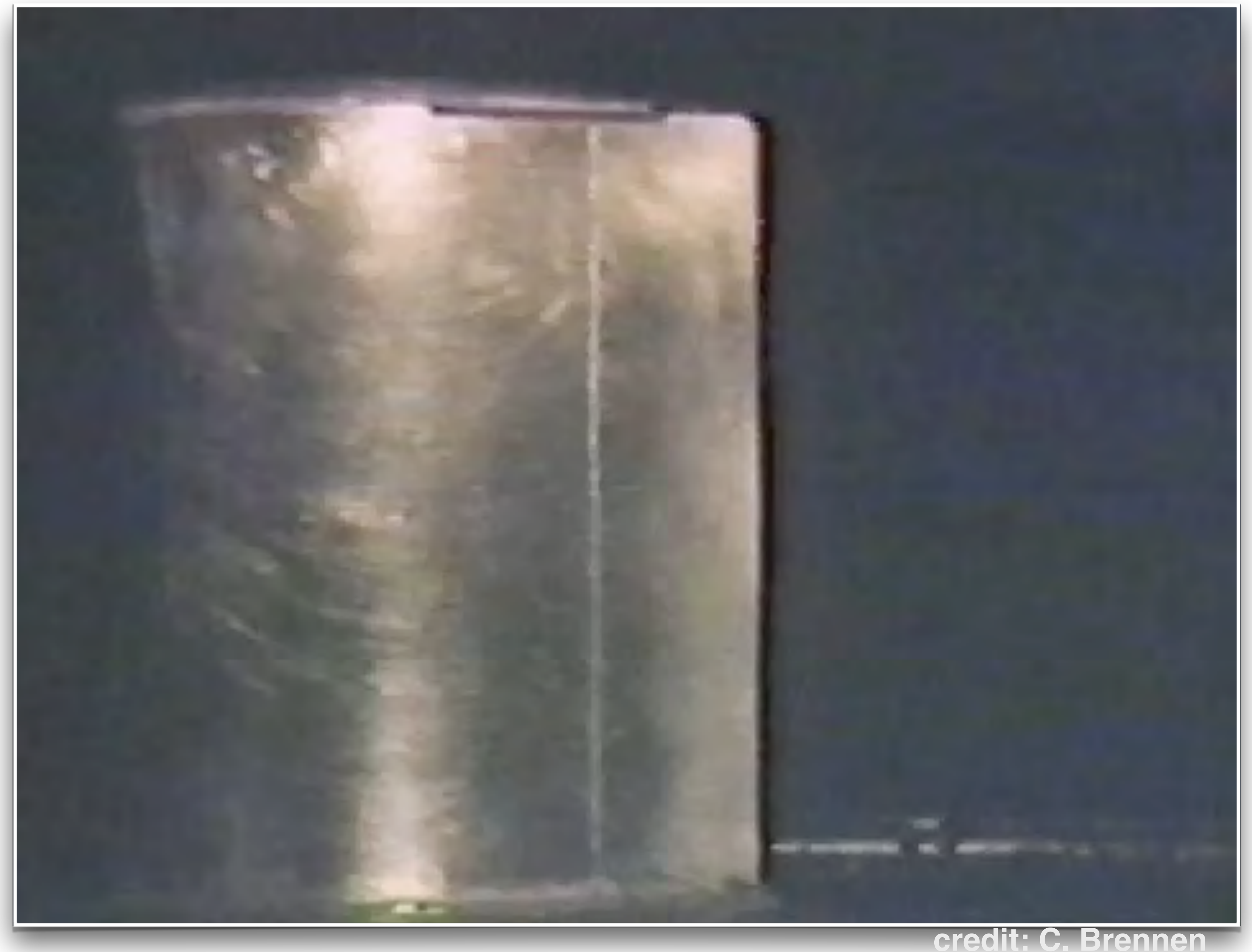
Describe phenomena with few **fundamental principles**.



TODAY:

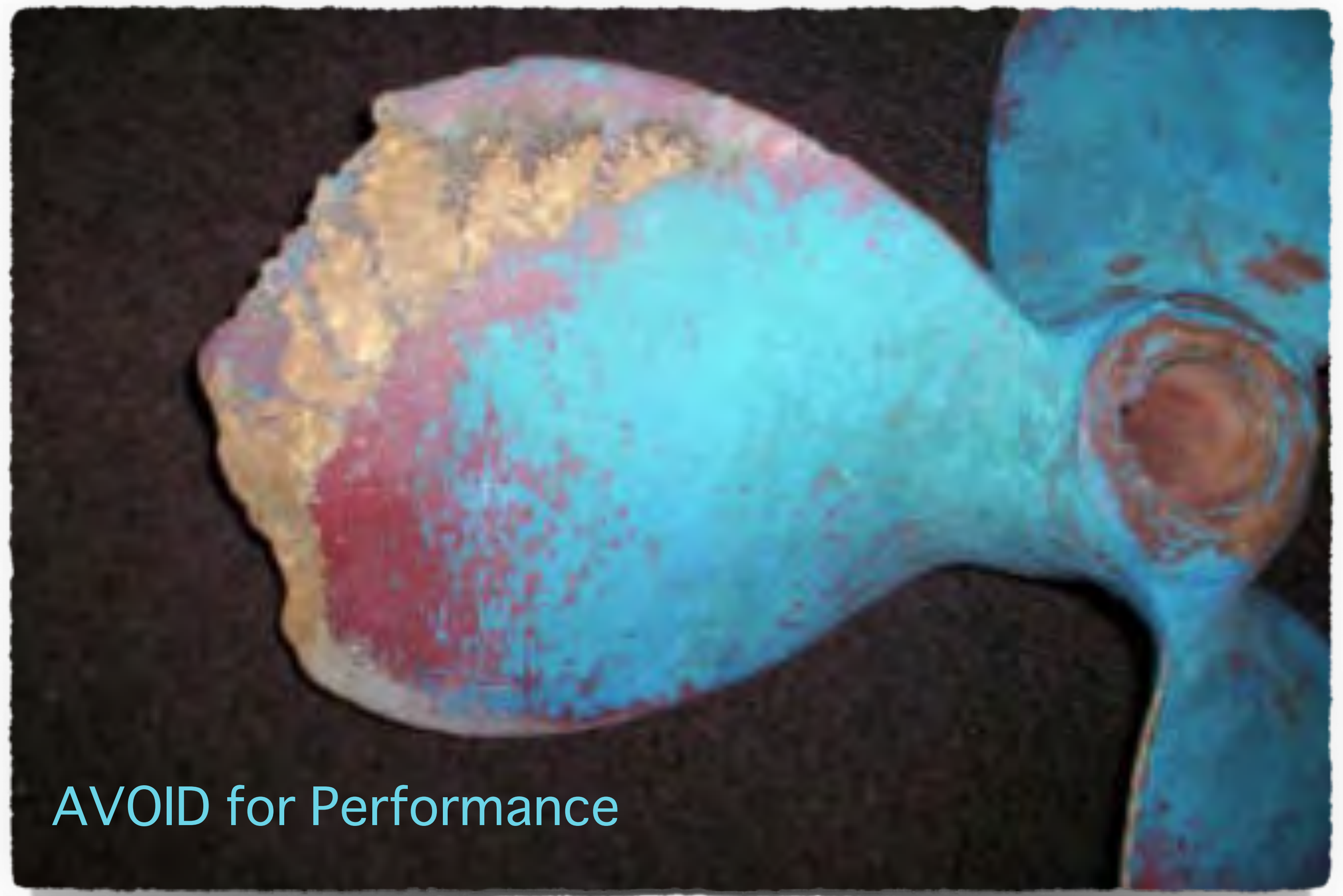
GREATLY ADVANCED BY COMPUTING

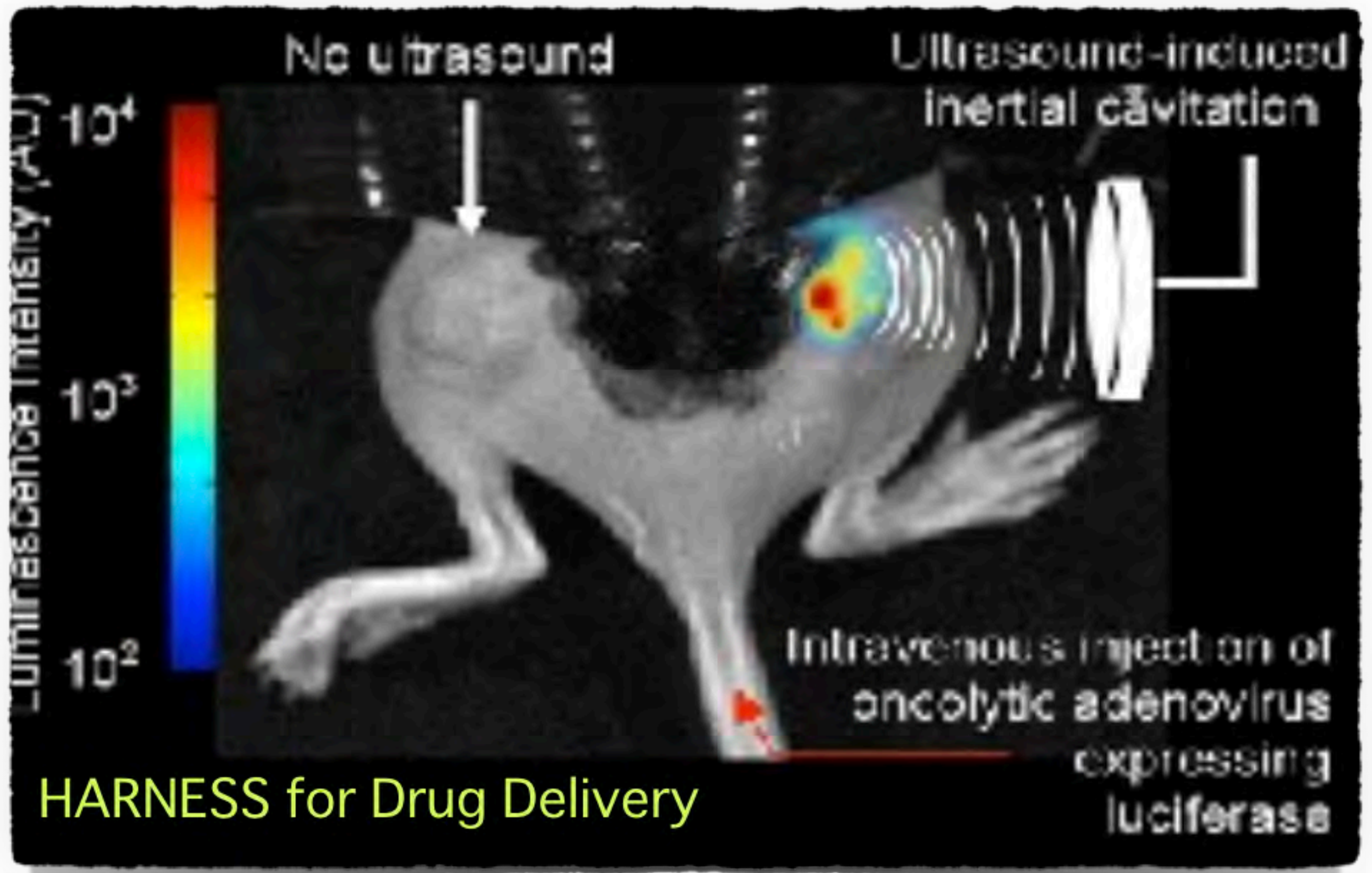
CASE STUDY: Cloud Cavitation Collapse



credit: C. Brennen

credit: DYNAFLOW





credit: C. Coussios - Oxford

GOVERNING EQUATIONS: Navier-Stokes and Interfaces

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}^T + p\mathbb{I}) = 0$$

$$\frac{\partial E}{\partial t} + \nabla \cdot ((E + p)\mathbf{u}) = 0$$

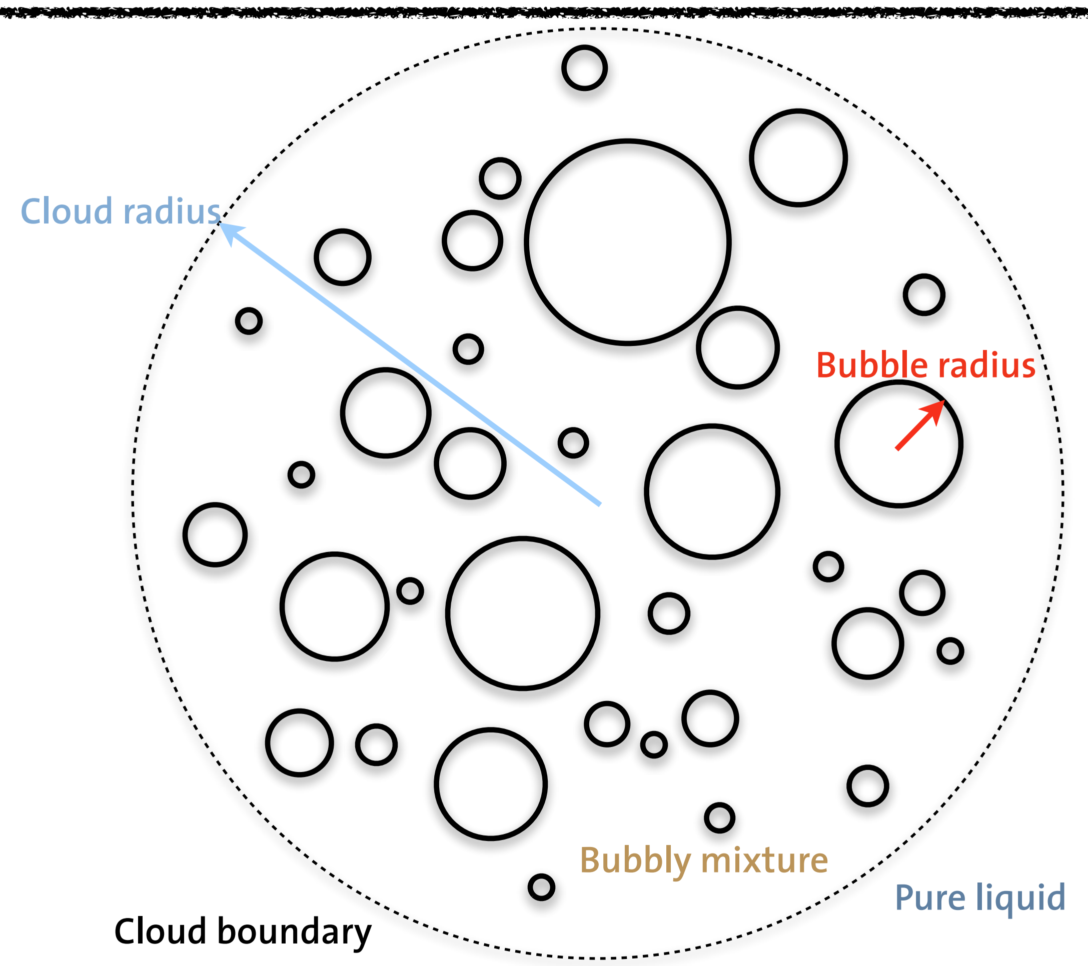
$$\frac{\partial \psi}{\partial t} + \mathbf{u} \cdot \nabla \psi = 0$$

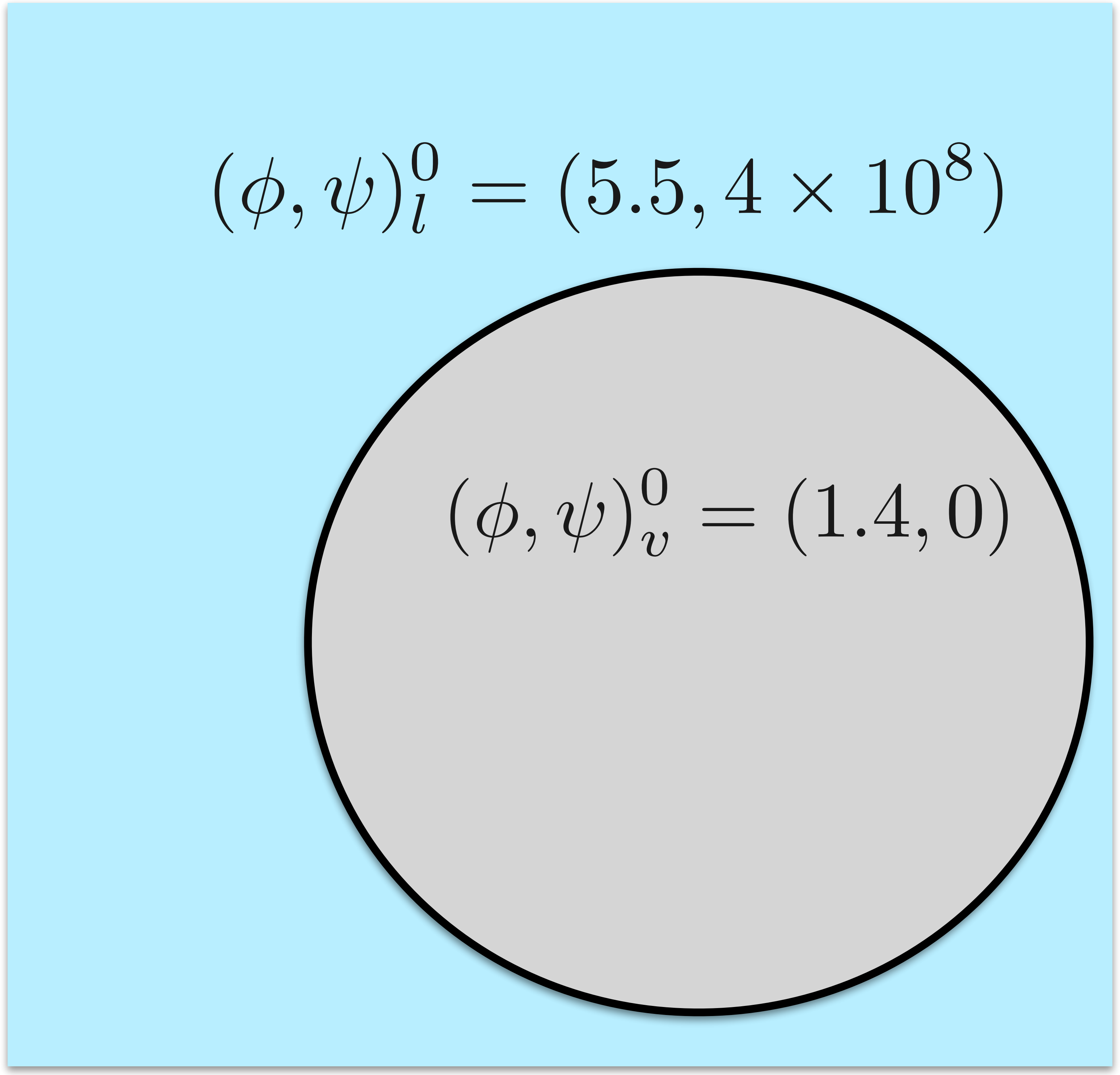
$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = 0$$

$$p = (\gamma - 1)(E - \frac{1}{2}\rho|\mathbf{u}|^2) - \Pi_\infty$$

$$\psi = \frac{\gamma \Pi_\infty}{\gamma - 1}$$

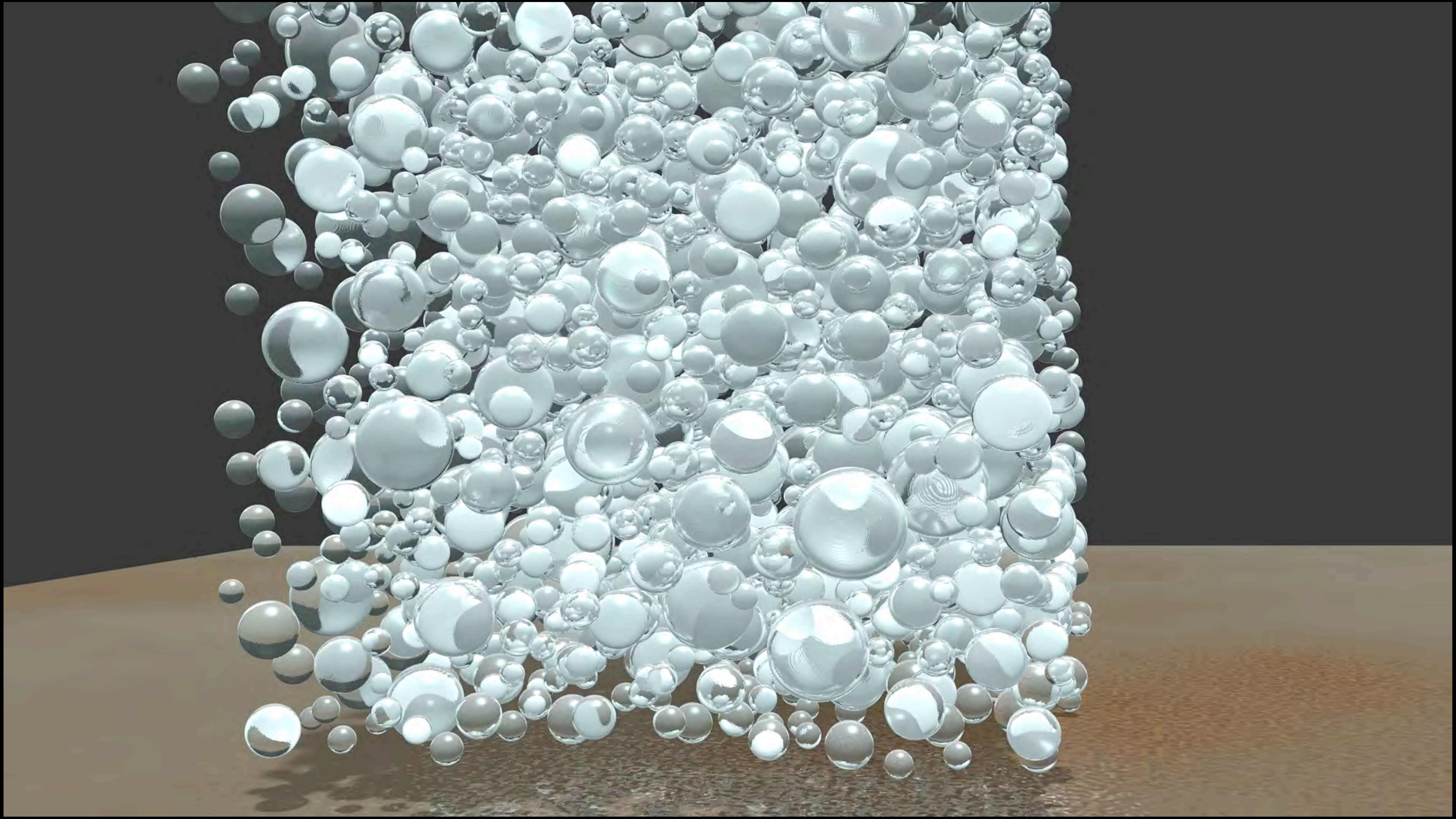
$$\phi = \frac{1}{\gamma - 1}$$





unknowns

$\rho, \mathbf{u}, E, p, \phi, \psi$



...complementing experiments and theory

Cavitation Collapse
as interacting micro-jets

EMPIRICAL KNOWLEDGE

"Useful" dependencies from data or derived from experience.

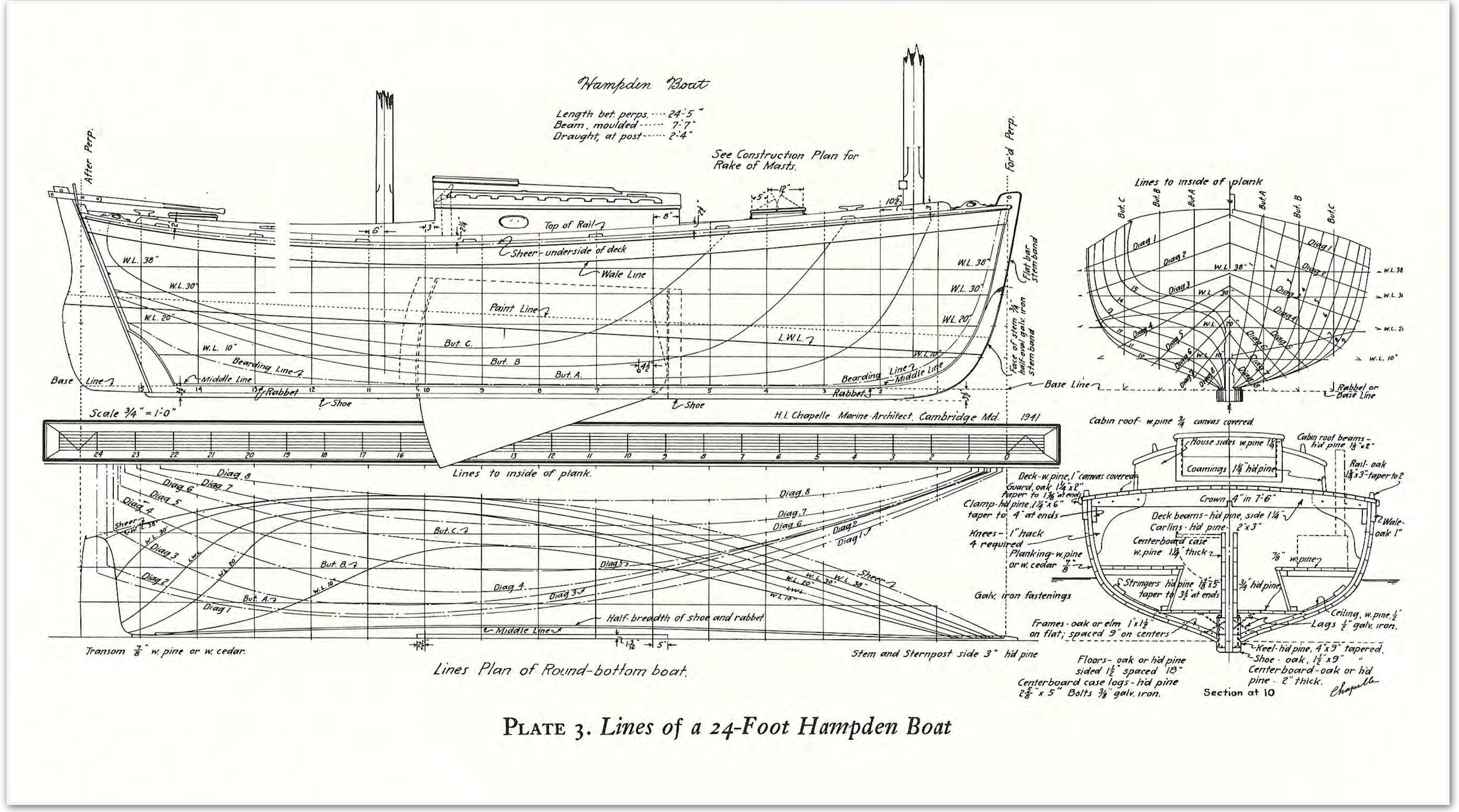


PLATE 3. Lines of a 24-Foot Hampden Boat

TODAY:

GREATLY ADVANCED BY COMPUTING

Bloomberg Markets

Markets Tech Pursuits Politics Opinion Businessweek

JPMorgan Software Does in Seconds What Took Lawyers 360,000 Hours

by **Hugh Son**

February 27, 2017, 7:31 PM EST *Updated on* February 28, 2017, 7:24 AM EST

- New software does in seconds what took staff 360,000 hours
- Bank seeking to streamline systems, avoid redundancies

At JPMorgan Chase & Co., a learning machine is parsing financial deals that once kept legal teams busy for thousands of hours.
The program, called COIN, for Contract Intelligence, does the mind-numbing job of interpreting commercial-loan agreements that, until the project went online in June, consumed 360,000 hours of work each year by lawyers and loan officers. The software reviews documents in seconds, is less error-prone and never asks for vacation.

NATURE | LETTER

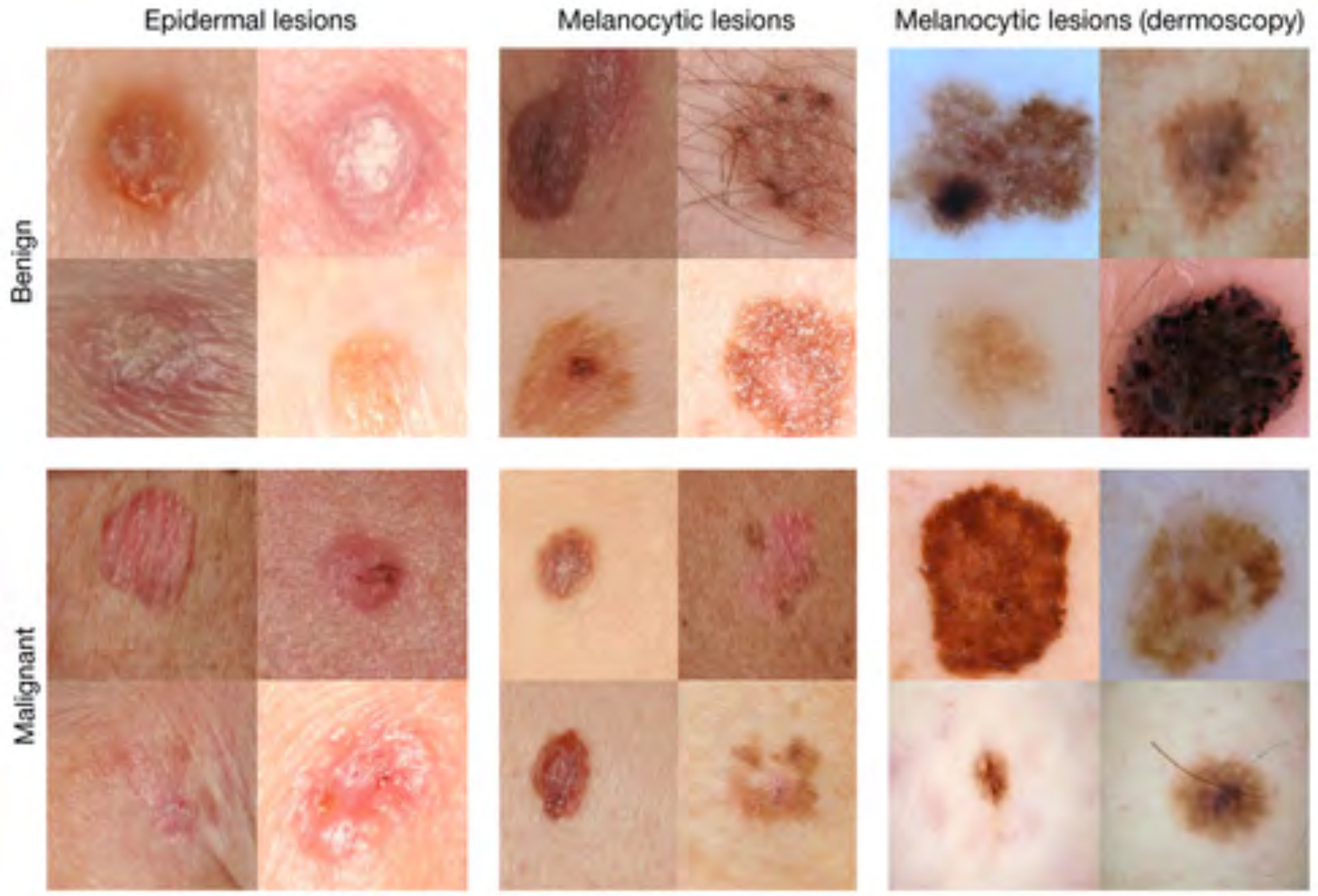
日本語要約

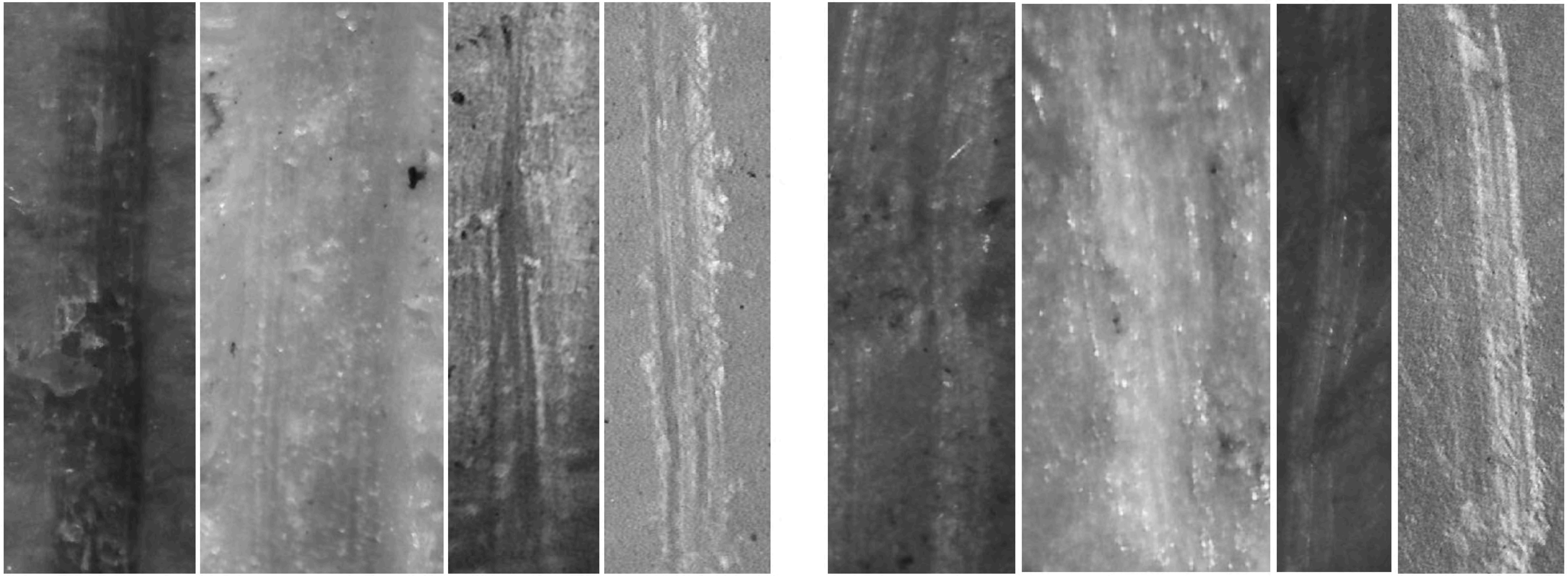
Dermatologist-level classification of skin cancer with deep neural networks

Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Susan M. Swetter, Helen M. Blau & Sebastian Thrun

Affiliations | Contributions | Corresponding authors

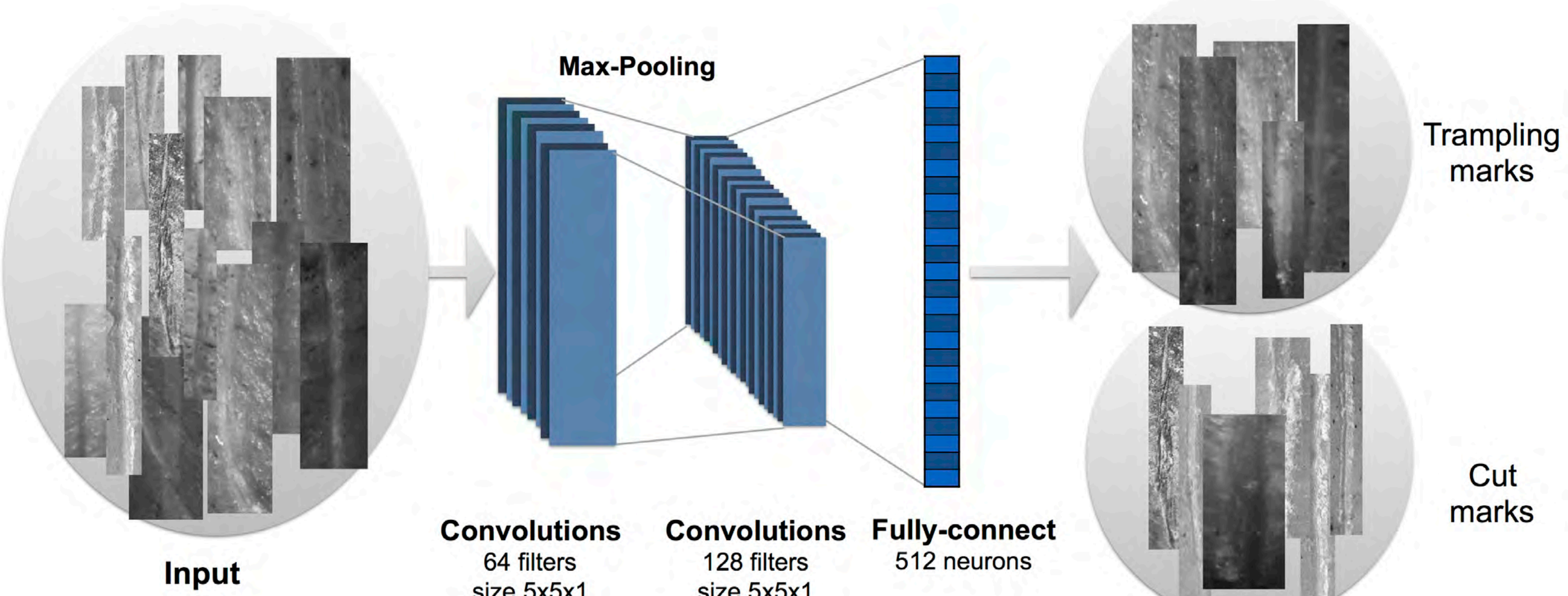
Nature **542**, 115–118 (02 February 2017) | doi:10.1038/nature21056
Received 28 June 2016 | Accepted 14 December 2016 | Published online 25 January 2017



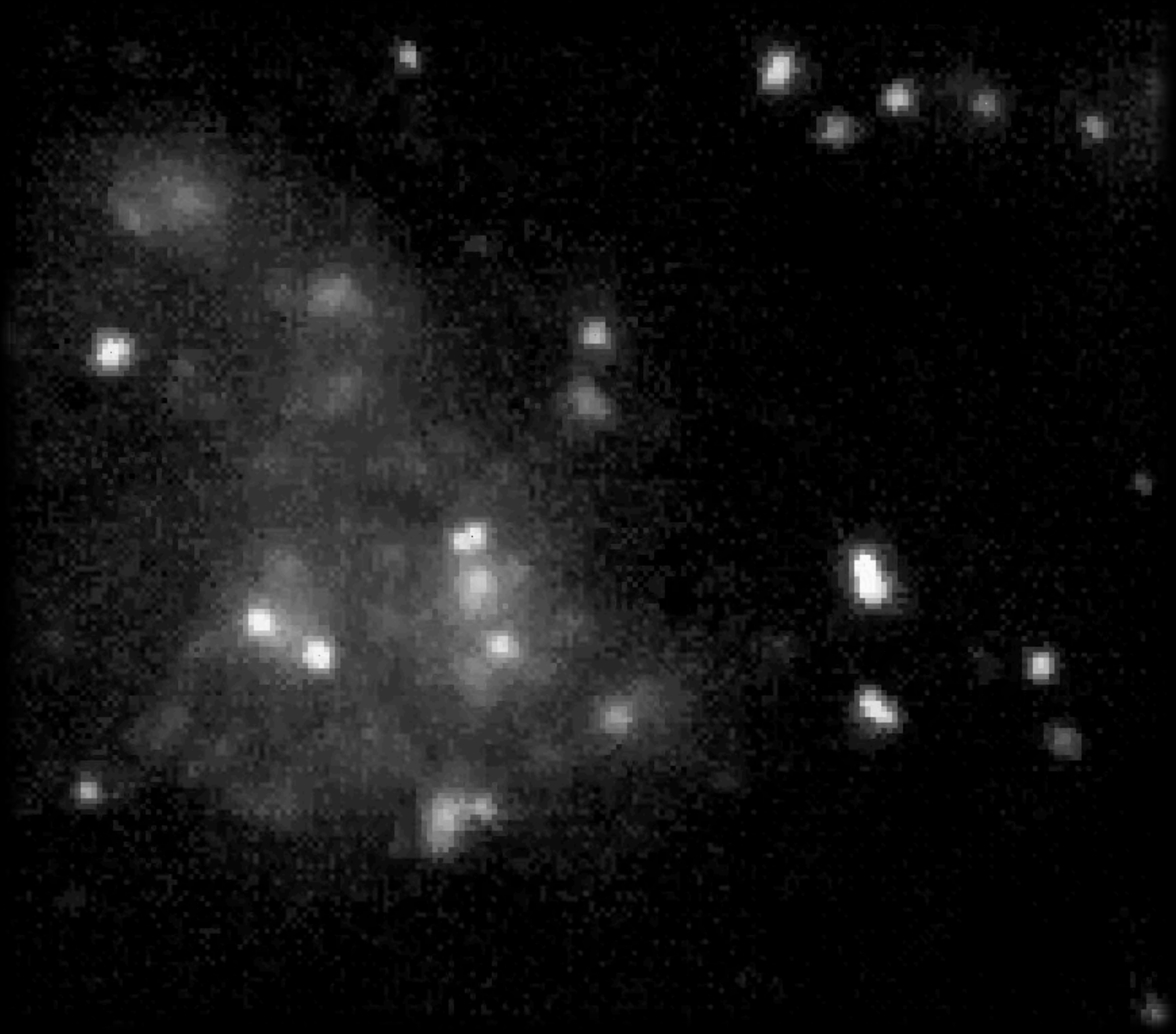


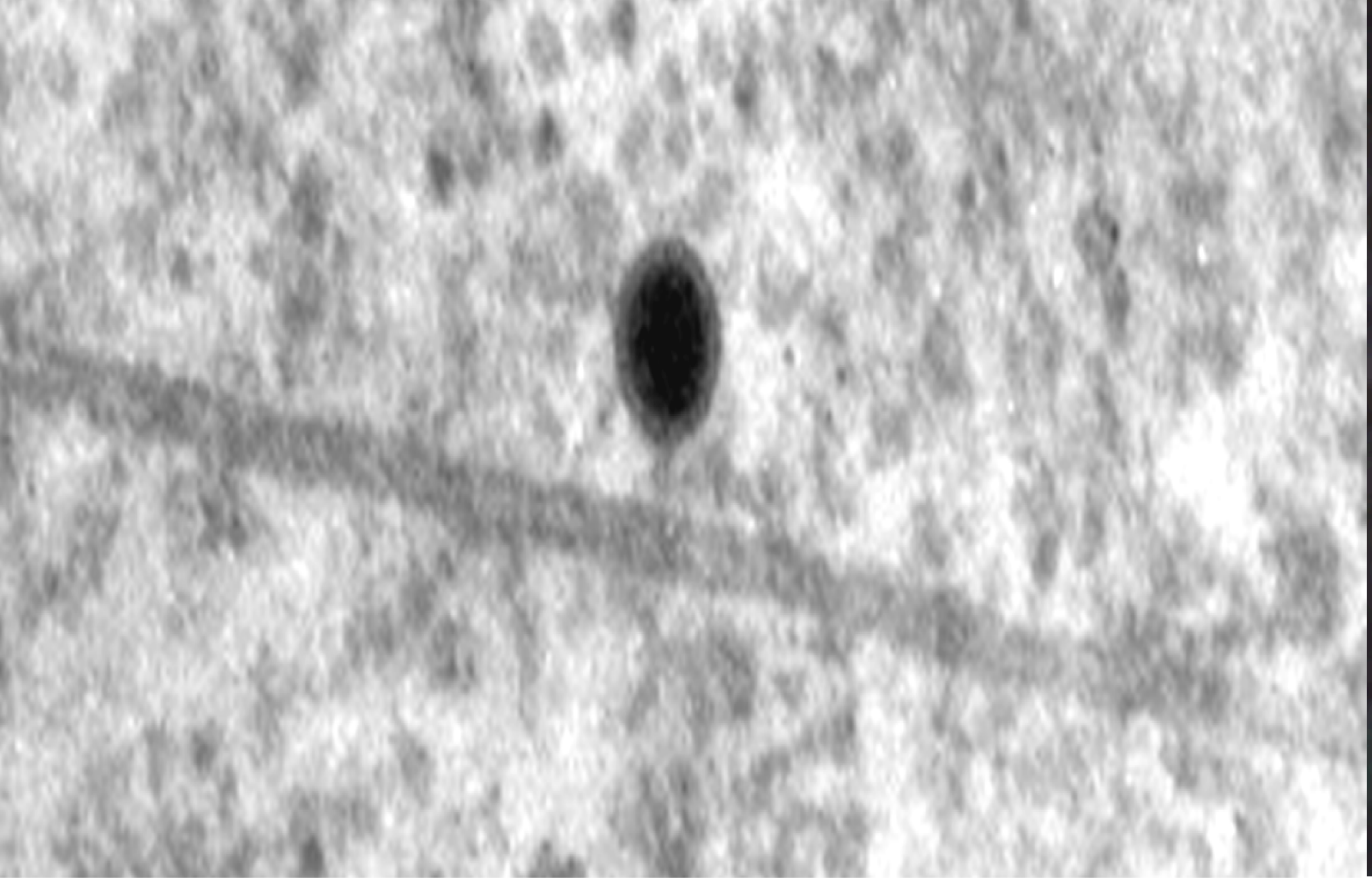
When did humans start hunting/eating meat ?

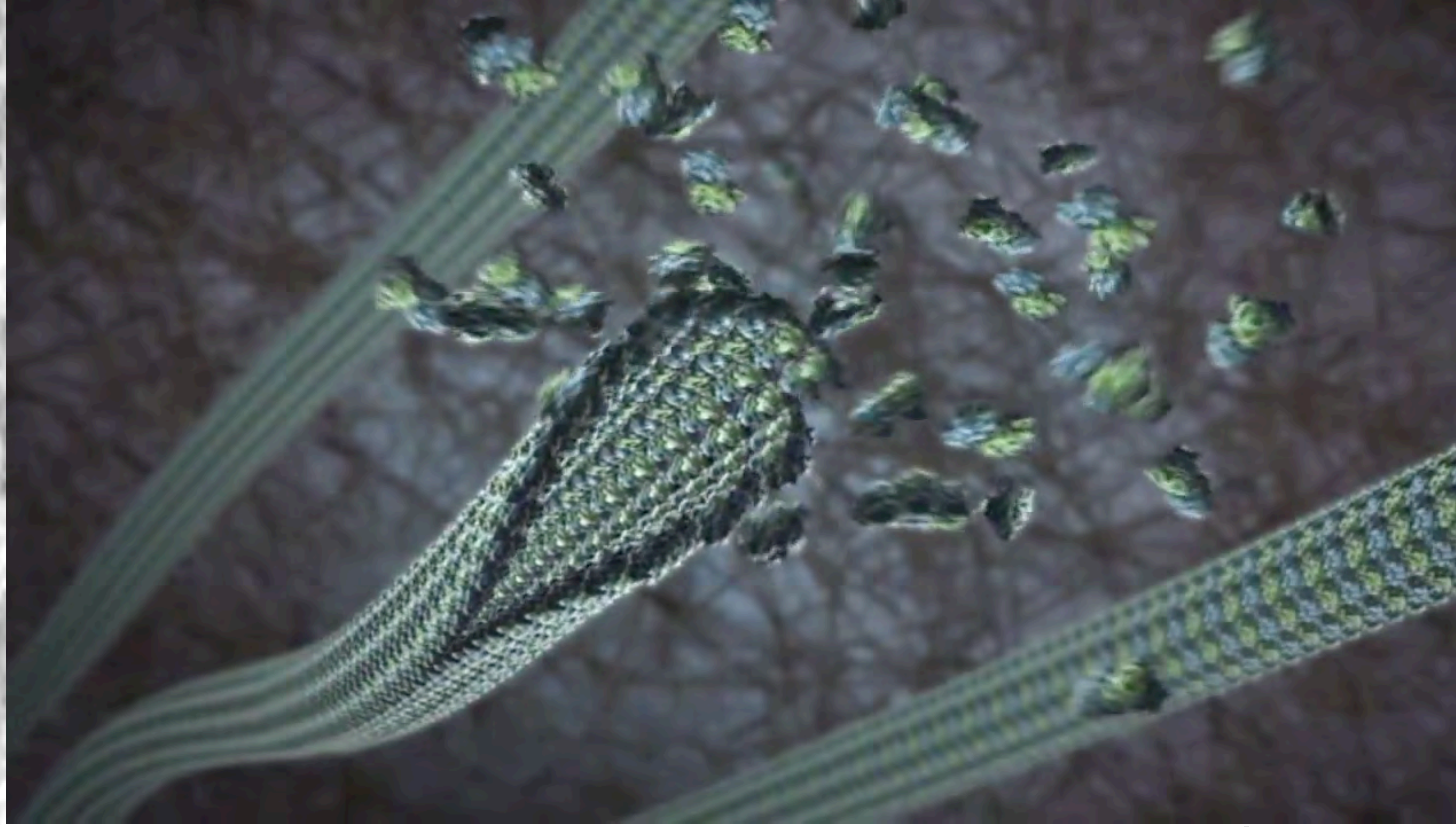
with: Manuel Domínguez-Rodrigo
Department of Prehistory
Complutense University, Madrid, Spain



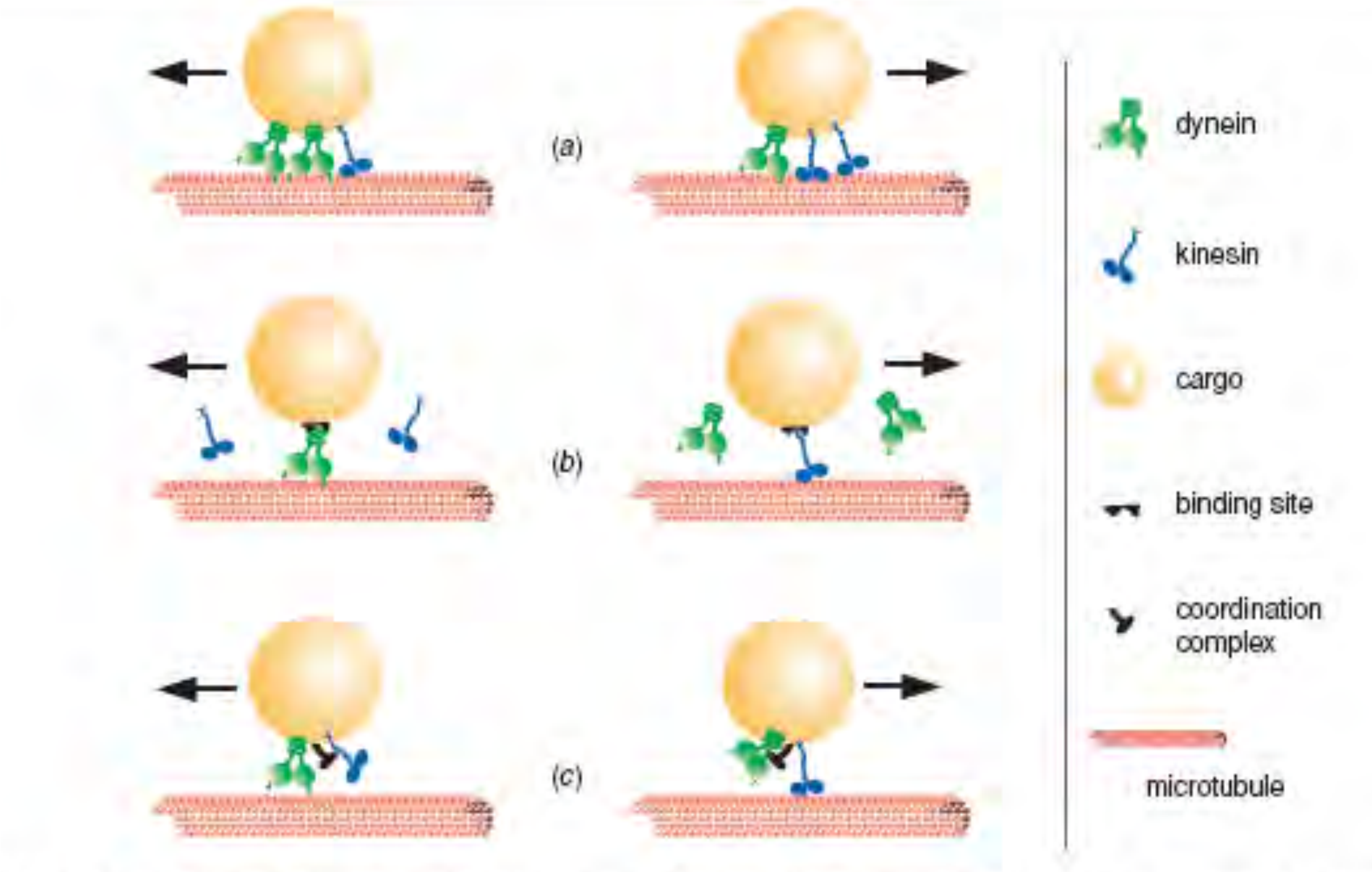
Single Virus Trajectory Segmentation







credit: Biovisions

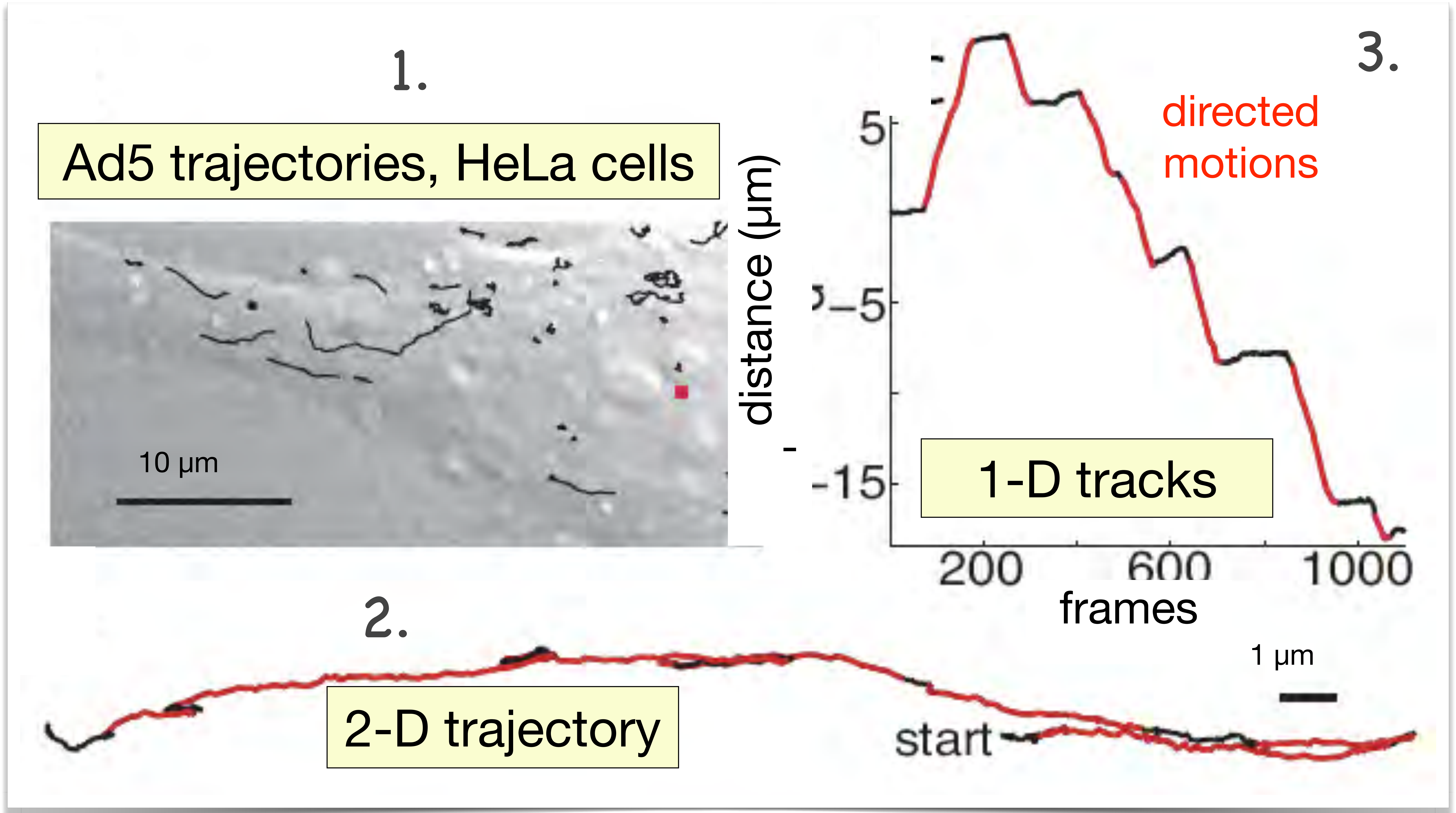


Virus Entry: Open Sesame

Mark Marsh[1,*] and Ari Helenius[2,*]
[1] Cell Biology Unit, MRC Laboratory for Molecular Cell Biology, and Department of Biochemistry and Molecular Biology, University College London, Gower Street, London WC1E 6BT, United Kingdom
[2] Institute of Biochemistry, ETH Zurich, ETH Hönggerberg HPM E 6.3, CH-8093 Zurich, Switzerland
*Contact: m.marsh@ucl.ac.uk (M.M.); ari.helenius@bc.biol.ethz.ch (A.H.)
DOI 10.1016/j.cell.2006.02.007

Detailed information about the replication cycle of viruses and their interactions with host organisms is required to develop strategies to stop them. Cell biology studies, live-cell imaging, and systems biology have started to illuminate the multiple and subtly different pathways that animal viruses use to enter host cells. These insights are revolutionizing our understanding of endocytosis and the movement of vesicles within cells. In addition, such insights reveal new targets for attacking viruses before they can usurp the host-cell machinery for replication.

Directed motions -> segmented 1-D tracks

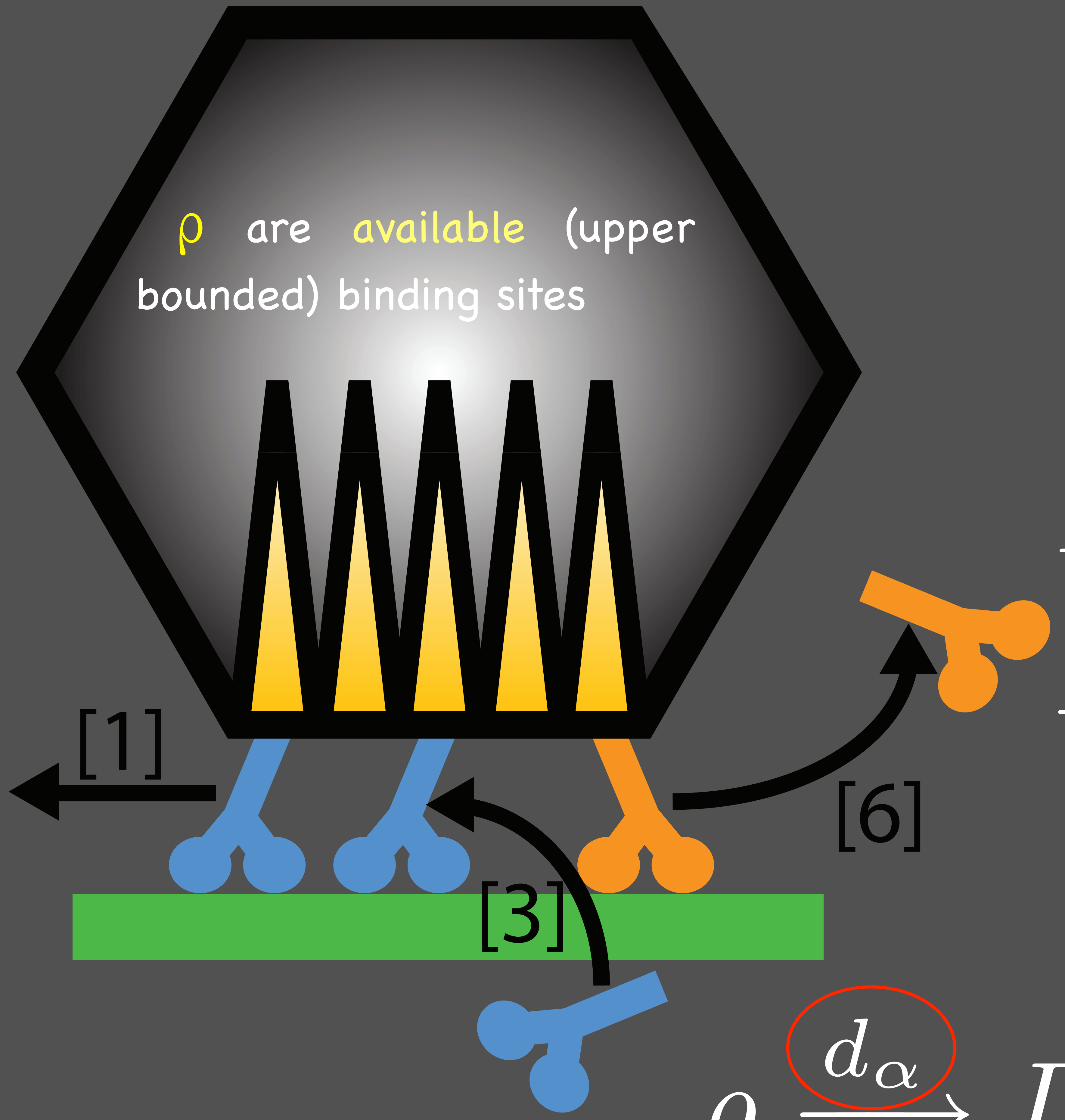


Helmuth J.A., Burckhardt Ch.J., Koumoutsakos P., Greber U.F., Sbalzarini I.,, **J. Structural Biology,** 159, 3, 347-358, 2007

Model for Virus Motion

Dynein (D) / Kinesin (K) :

MOVE - BIND - UNBIND



MOVE

$$D \xrightarrow{d_\mu} D_{\text{moves}}$$

$$K \xrightarrow{k_\mu} K_{\text{moves}}$$

UN-BIND

$$K \xrightarrow{k_\delta} \rho$$

$$D \xrightarrow{d_\delta} \rho$$

BIND

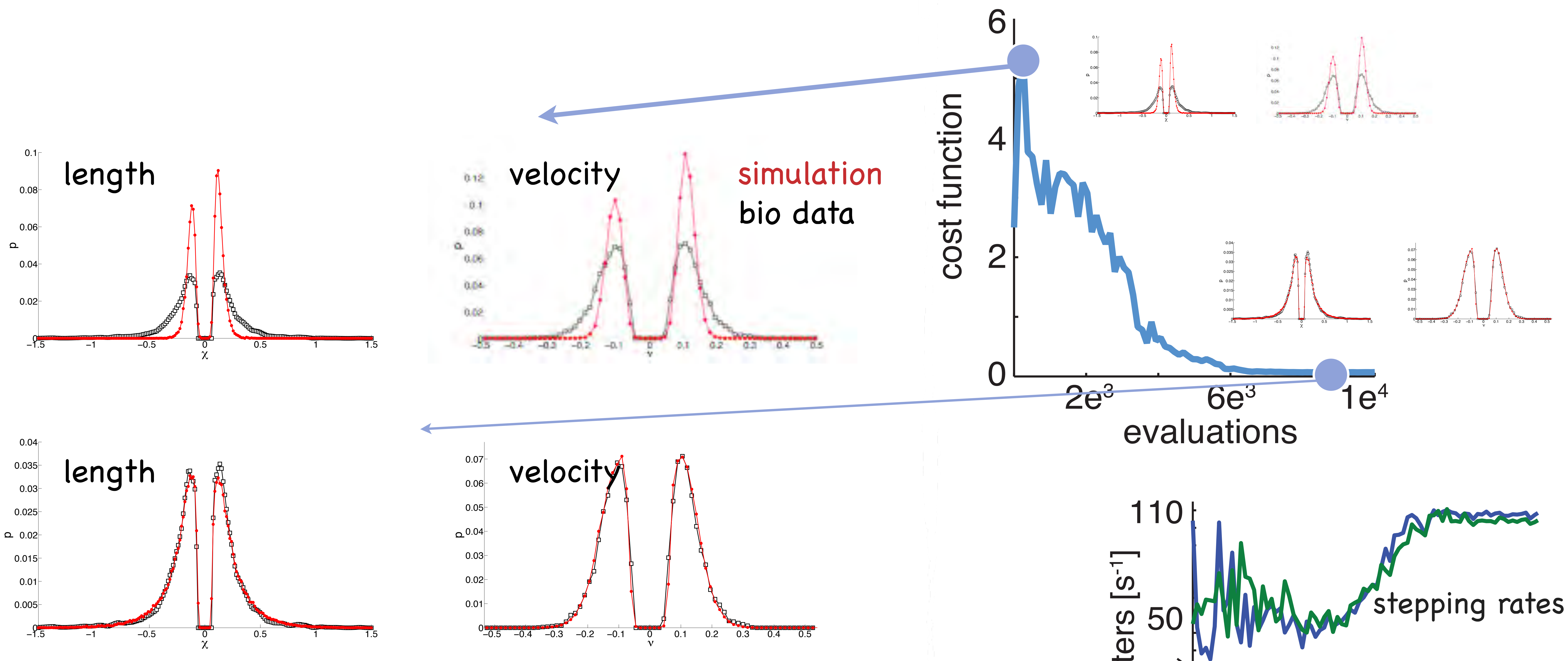
$$\rho \xrightarrow{d_\alpha} D$$

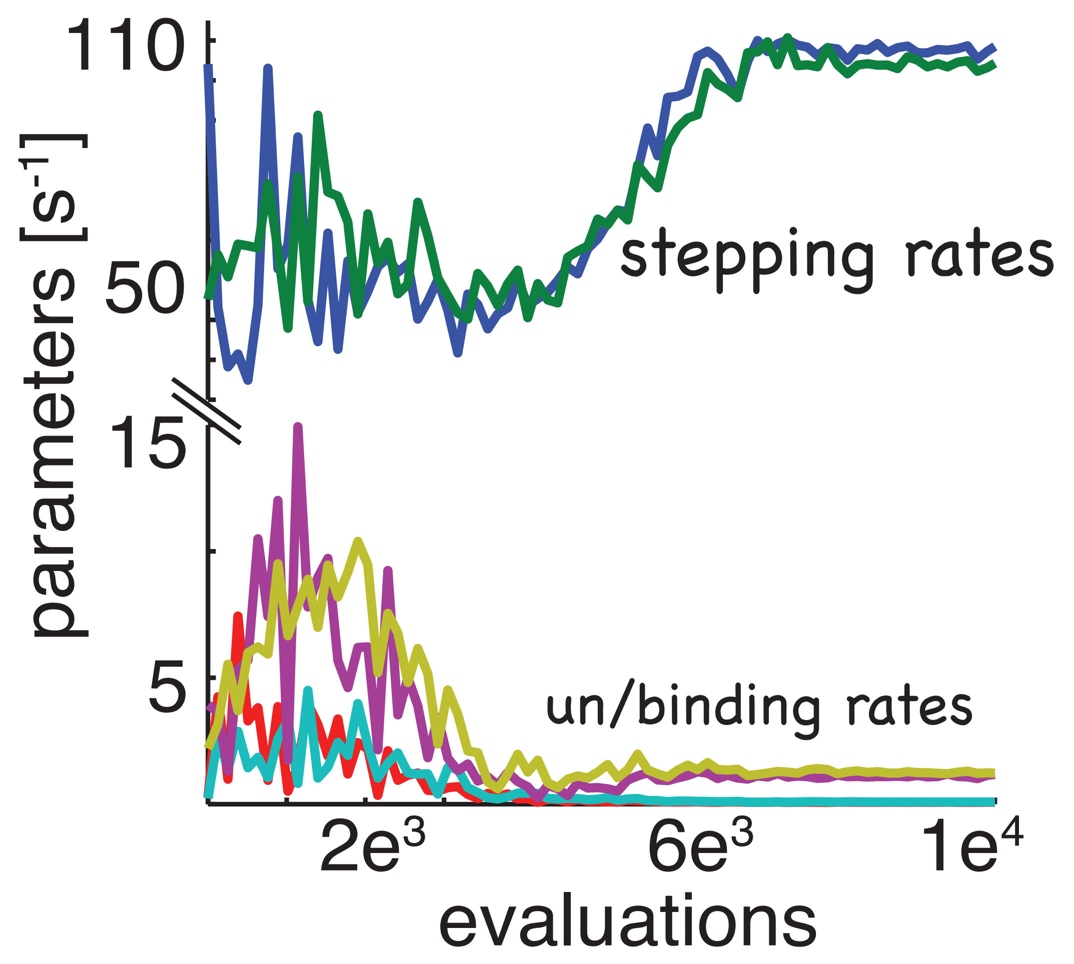
$$\rho \xrightarrow{k_\alpha} K$$

d_μ, k_μ :displacement rates for dynein and kinesin,

d_α, d_δ : dynein binding and unbinding rates.

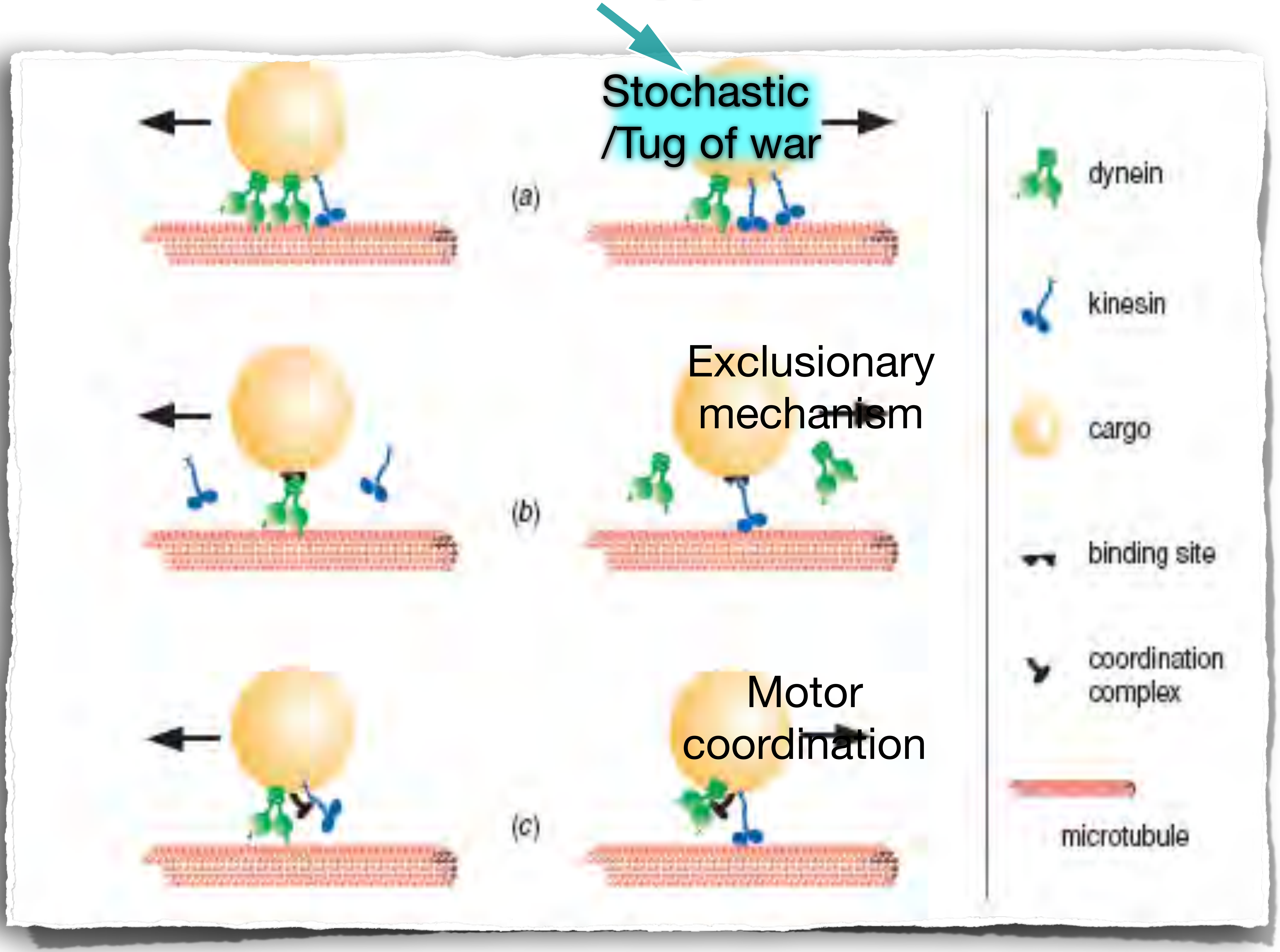
Optimization





- Single motor velocities ~850nm/s
- Binding/unbinding rate ~ 10^{-2} x stepping rate
- High sensitivity on binding/unbinding rates

Model suggests...



from Gross, Physical Biology (2004)

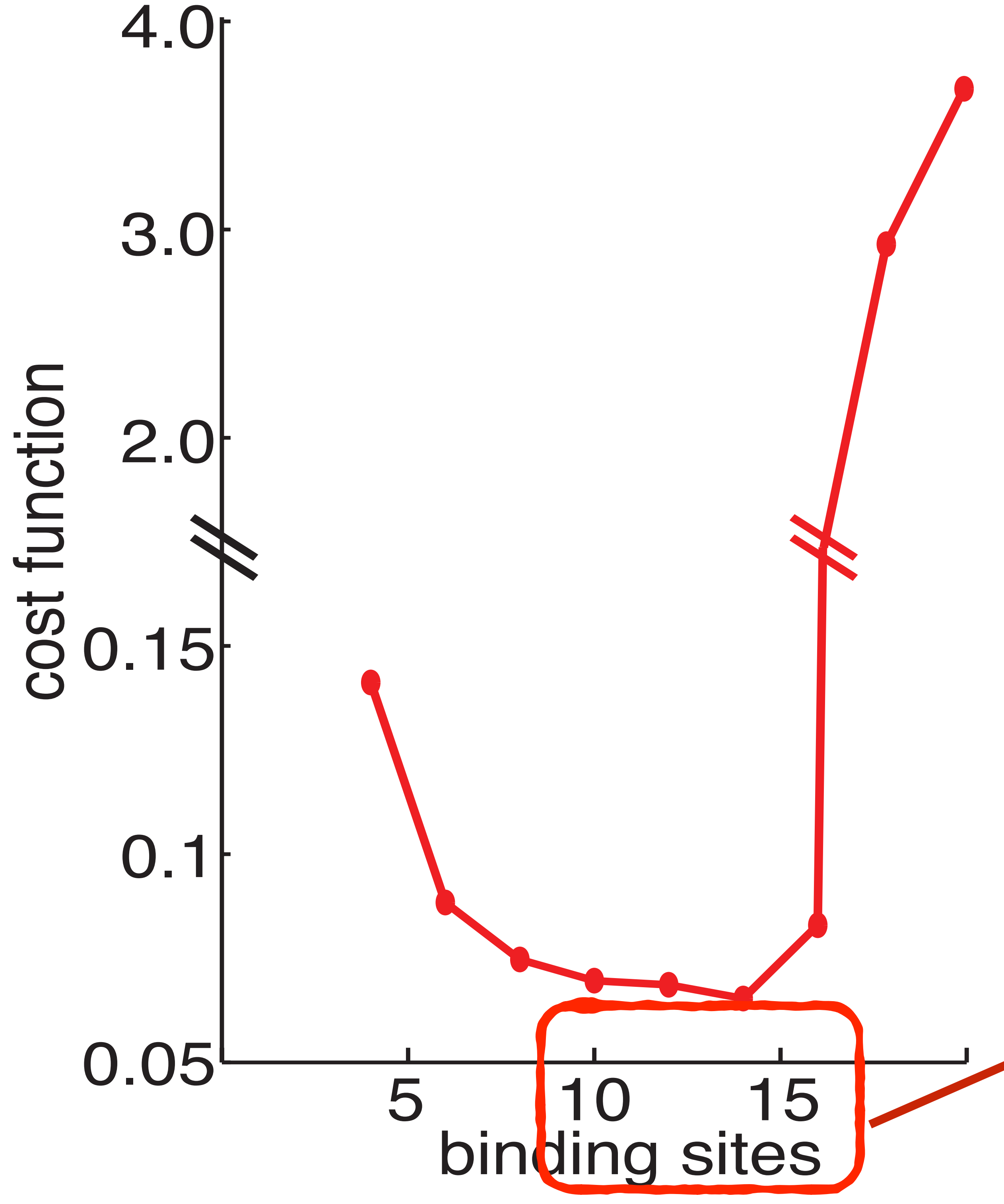
Model predicts an non-coordinated ‘tug-of-war’

- Distinct or overlapping binding sites for dynein & kinesin are possible
- k_{on} and k_{off} for dynein and kinesin are lower than expected
- 1-3 motors bound per virus during runs
- a total of 8-14 motor binding sites required

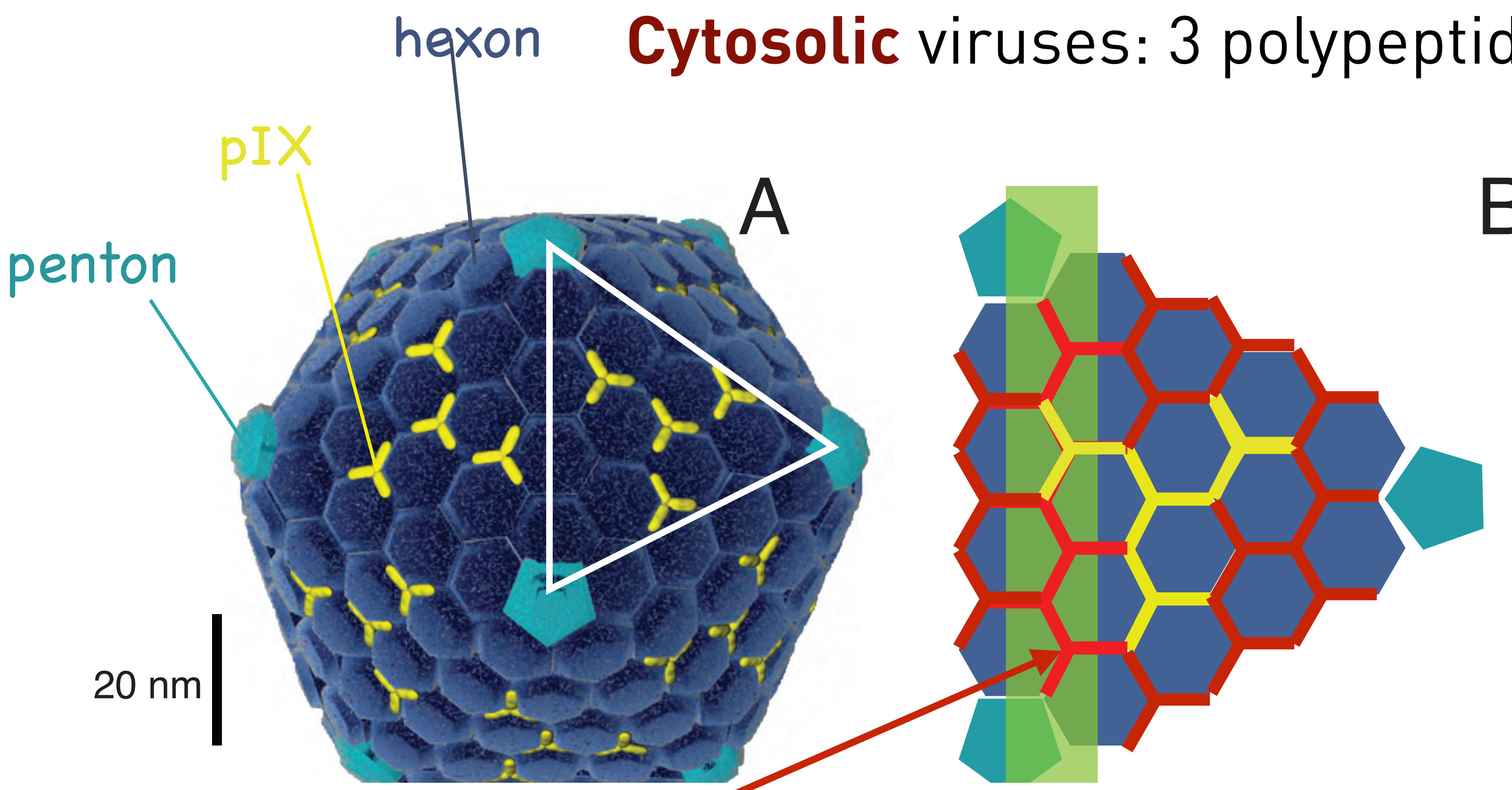
WHERE ARE THE **BINDING SITES ?**

Virus=90nm (90 nm in diameter) - Microtubule = 25 nm

Postulate that the maximum number of microtubule motor-capsid interactions occurs along the edge of a capsid facet, in this case on hexon



Cytosolic viruses: 3 polypeptides present



Cell Host & Microbe
Article

Adenovirus Transport via Direct Interaction of Cytoplasmic Dynein with the Viral Capsid Hexon Subunit

K. Helen Bremner,[1] Julian Scherer,[1,3] Julie Yi,[1,3] Michael Vershinin,[2,3] Steven P. Gross,[2] and Richard B. Vallee[1,*]

[1]Department of Pathology and Cell Biology, Columbia University, New York, NY 10032, USA

[2]Department of Developmental and Cell Biology, University of California at Irvine, Irvine, CA 92697, USA

[3]These authors contributed equally to this work

*Correspondence: rv2025@columbia.edu

DOI 10.1016/j.chom.2009.11.006

Candidate binding sites

1.hexon-hexon interfaces

[redacted]

CLASSICAL and EMPIRICAL KNOWLEDGE

- **Precise** predictions with explanations
- Few and **expensive** data/observations.
- **Possibly not "useful"** for applications

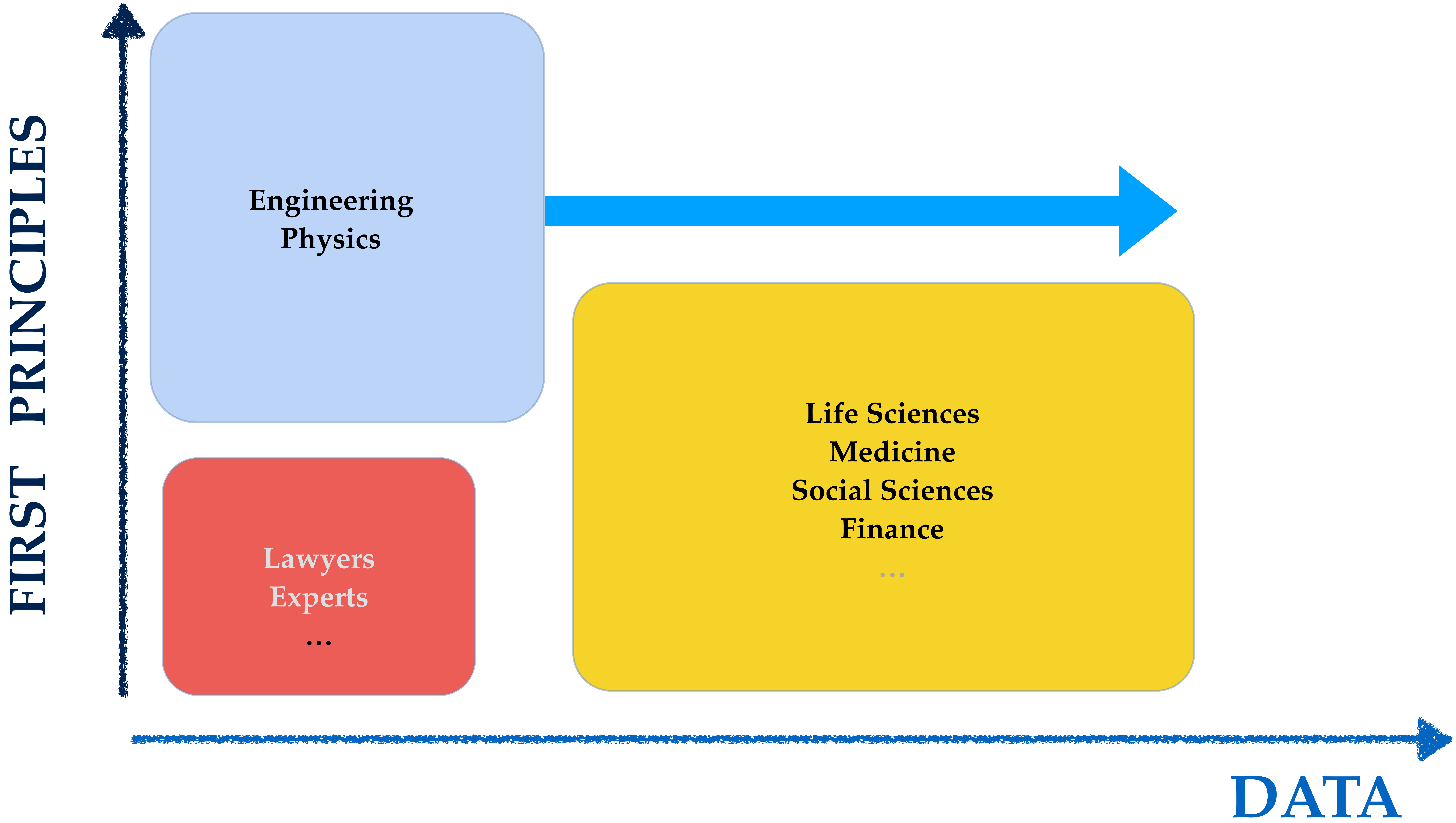
C
O
M
P
U
T
E
R

- Empirical Designs/**Statistical** predictions
- Relies on **"inexpensive"** data/observations
- **Practical utility** for a given application

CAUSALITY & HEURISTICS

IMPERFECTIONS & UNCERTAINTY

SOLVING PROBLEMS

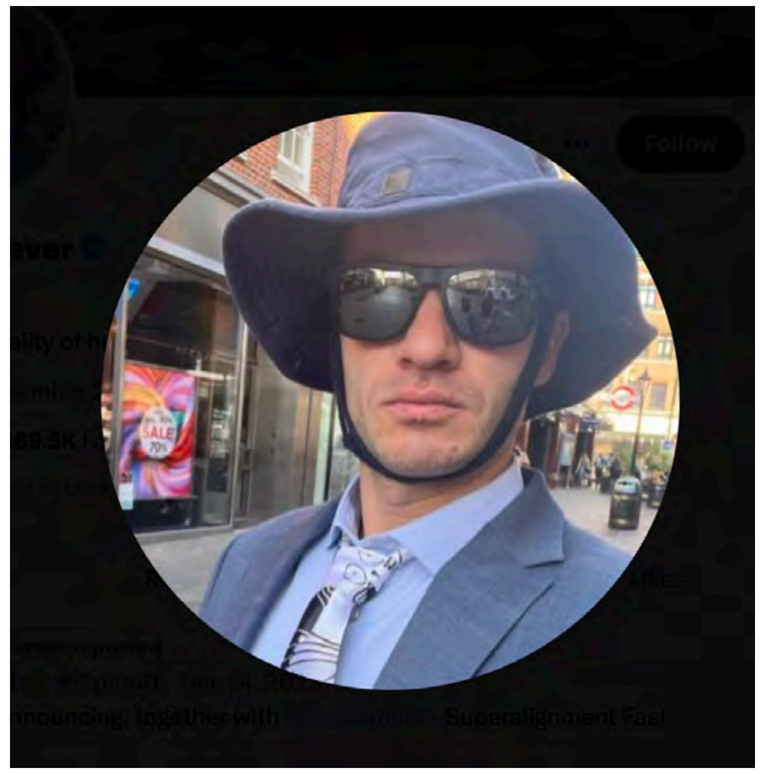


How to solve hard problems?

Use lots of training data.

And a big deep neural network.

And success is the only possible outcome.



Ilya Sutskever (2015),
co-founder of OpenAI

Bommasani, Rishi, et al. "On the opportunities and risks of foundation models." *arXiv preprint arXiv:2108.07258* (2021)

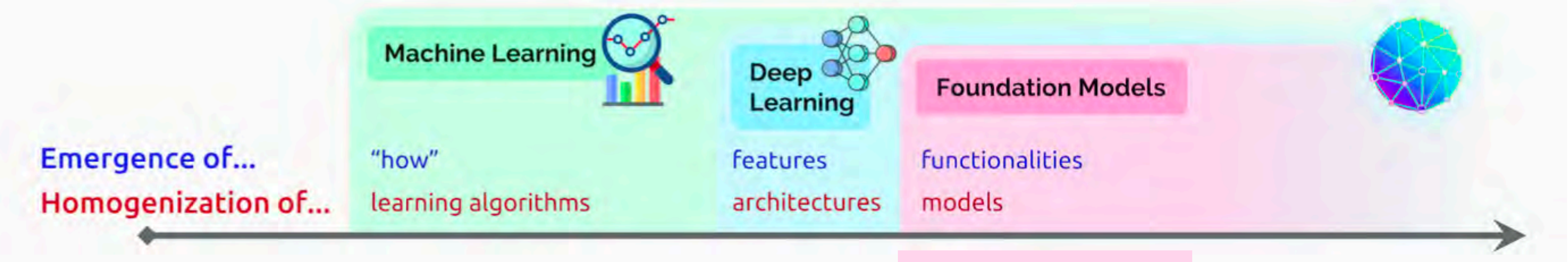
FOUNDATIONAL MODELS

Foundation models are AI neural networks trained on massive unlabeled datasets to handle a wide variety of jobs from translating text to analyzing medical images.

Bommasani, Rishi, et al. "On the opportunities and risks of foundation models." *arXiv preprint arXiv:2108.07258* (2021)

DL homogenizes model architectures

With **DL**, the **high-level features** used for prediction **emerge**

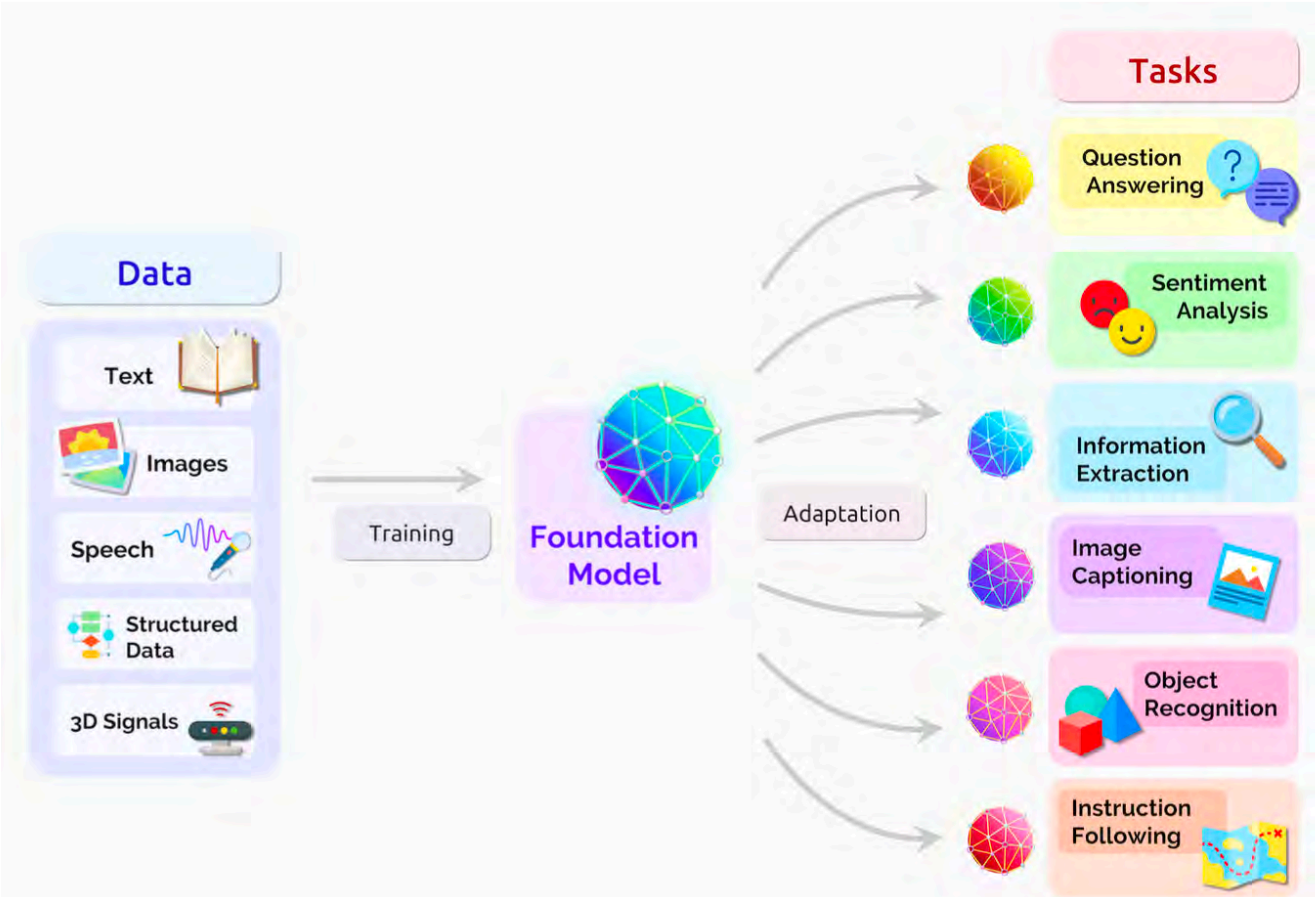


With **ML, how** a task is performed **emerges** from examples

with **FM**, **advanced functionalities** such as in-context learning **emerge**.

ML homogenizes learning algorithms

FM homogenizes the model itself (e.g., GPT-3)


Data
Text
Images
Speech
Structured Data
3D Signals
Training
Foundation Model
Adaptation
Tasks
Question Answering
Sentiment Analysis
Information Extraction
Image Captioning
Object Recognition
Instruction Following

EMERGENCE

First Principles -

EQUATION

+

CODE

+

PEOPLE

SCIENCE ADVANCES | RESEARCH ARTICLE

PHYSICS

Computing foaming flows across scales: From breaking waves to microfluidics

Petr Karnakov[1,2], Sergey Litvinov[1], Petros Koumoutsakos[1,2]*

HOMOGEN
IZATION

PARTICLES

+

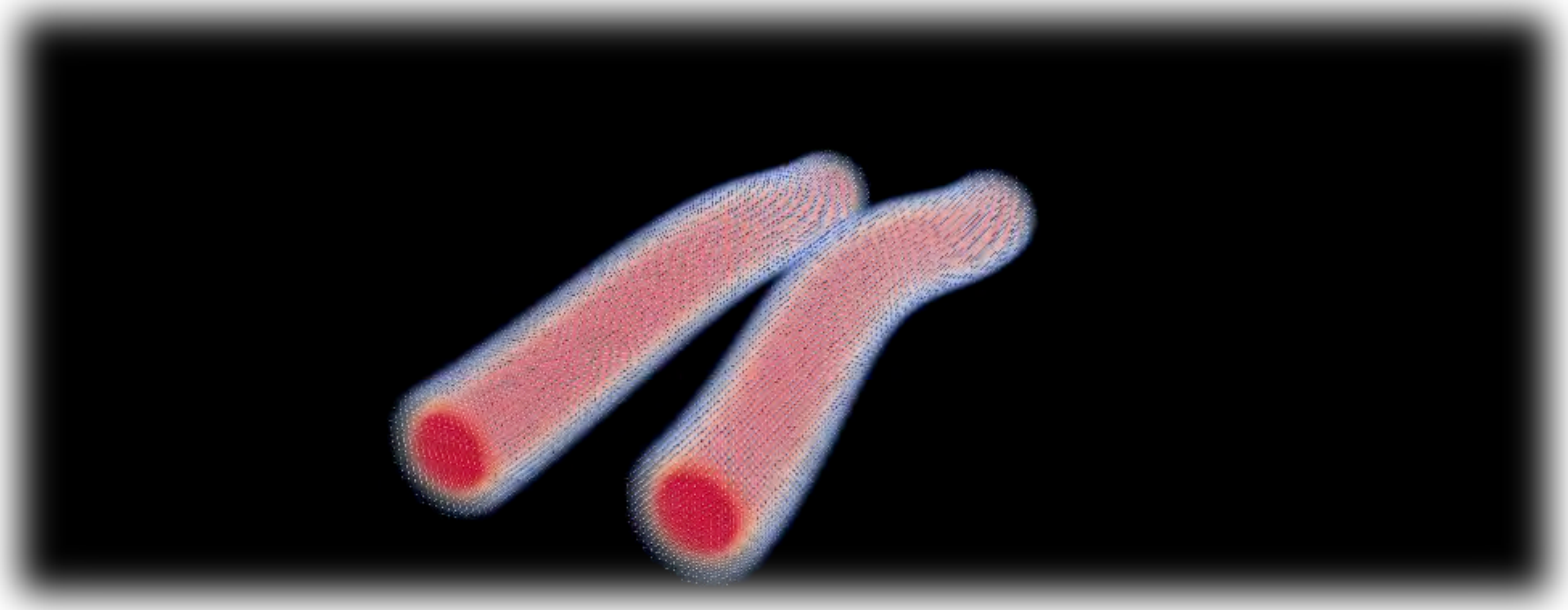
INTERACTIONS

PARTICLES & NEWTON'S LAWS

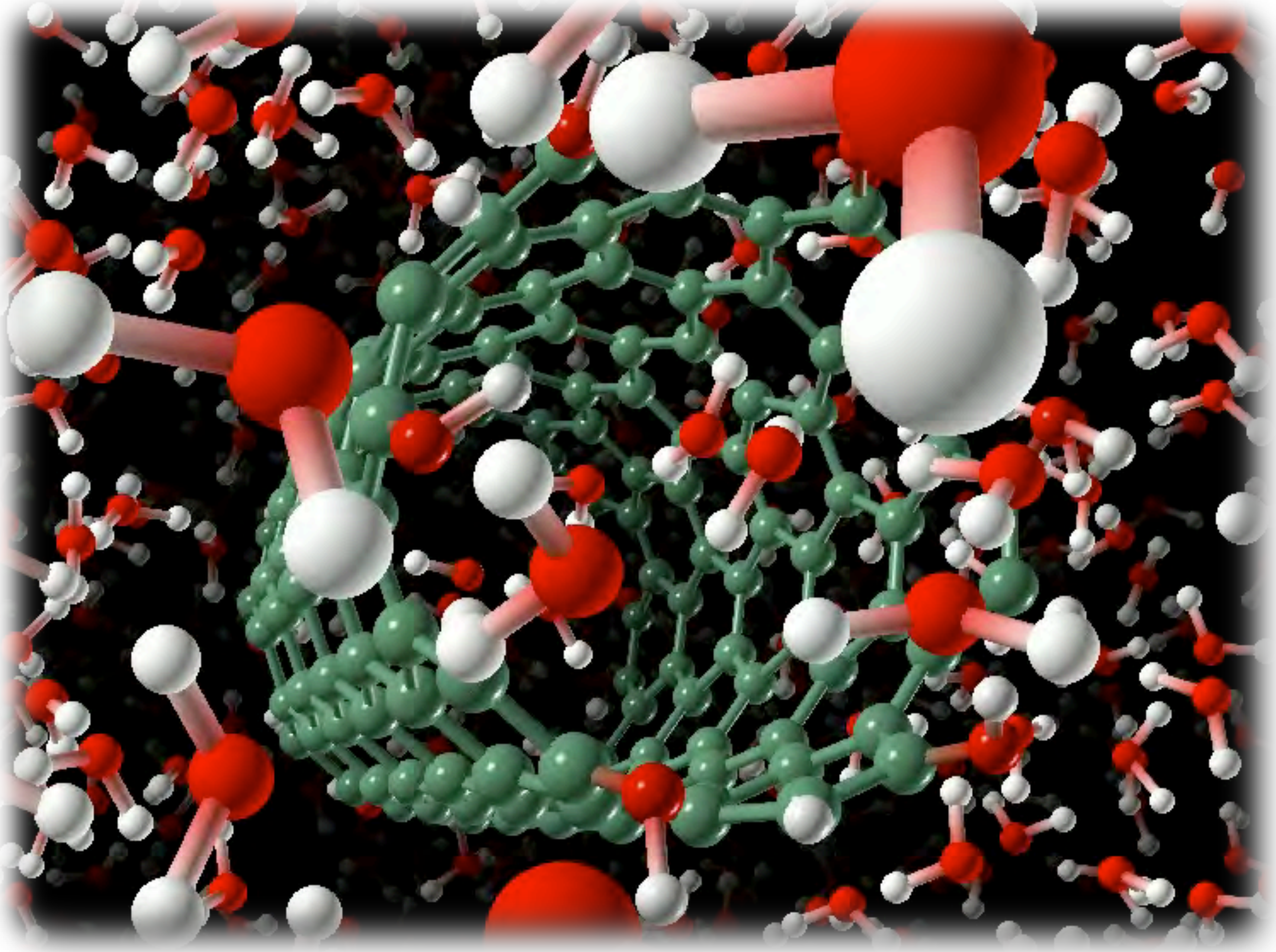
P. Koumoutsakos, Multiscale flow simulations using particles, **Ann. Rev. Fluid Mech.**, 2005

Vortex Methods, SPH

$$\rho_p \frac{D\mathbf{u_p}}{Dt} = (\nabla \cdot \sigma)_p$$



$$\frac{d\mathbf{x_p}}{dt} = \mathbf{u}_p$$



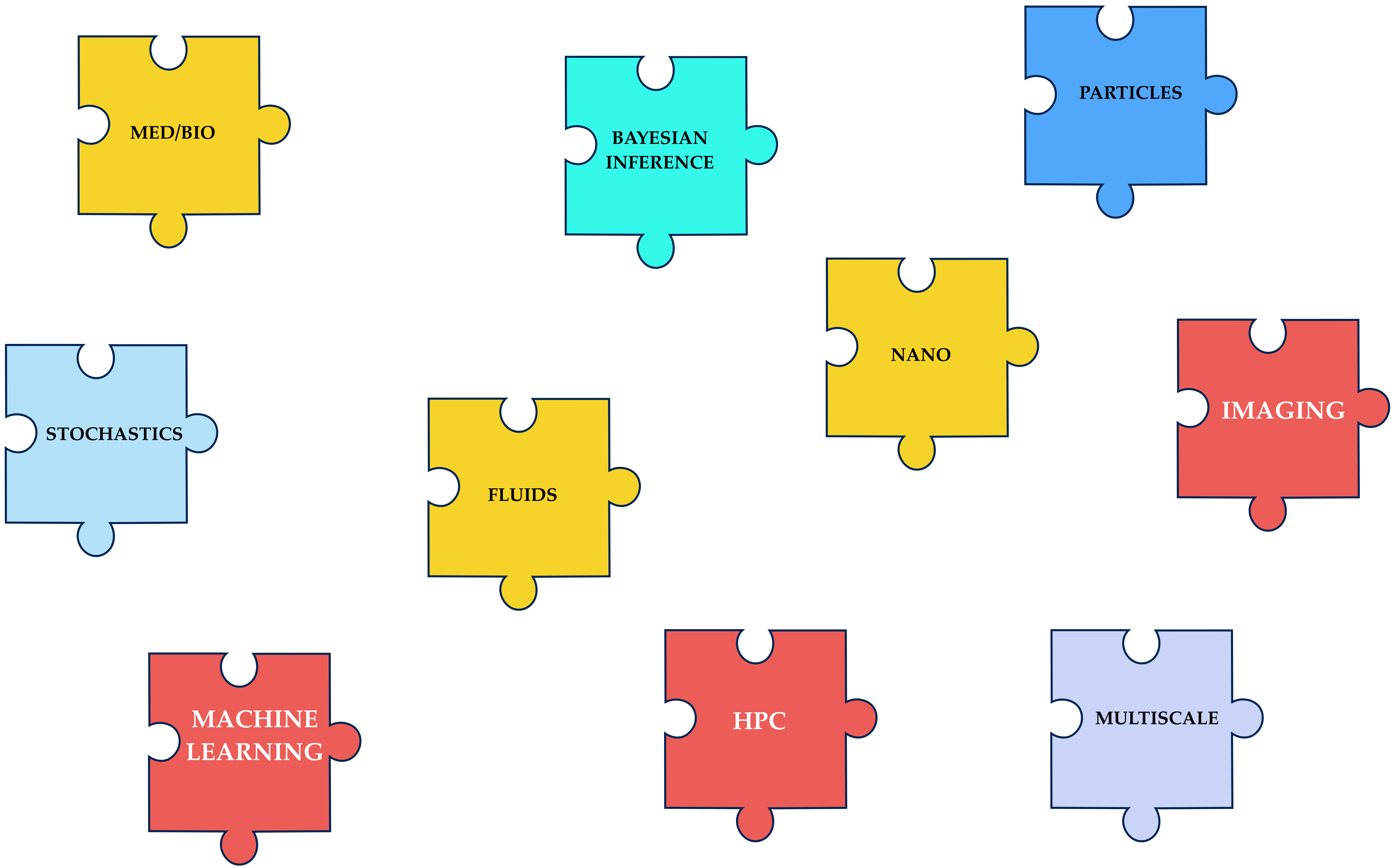
$$m\frac{d\mathbf{u_p}}{dt} = F_p$$

MD, DPD, CGMD

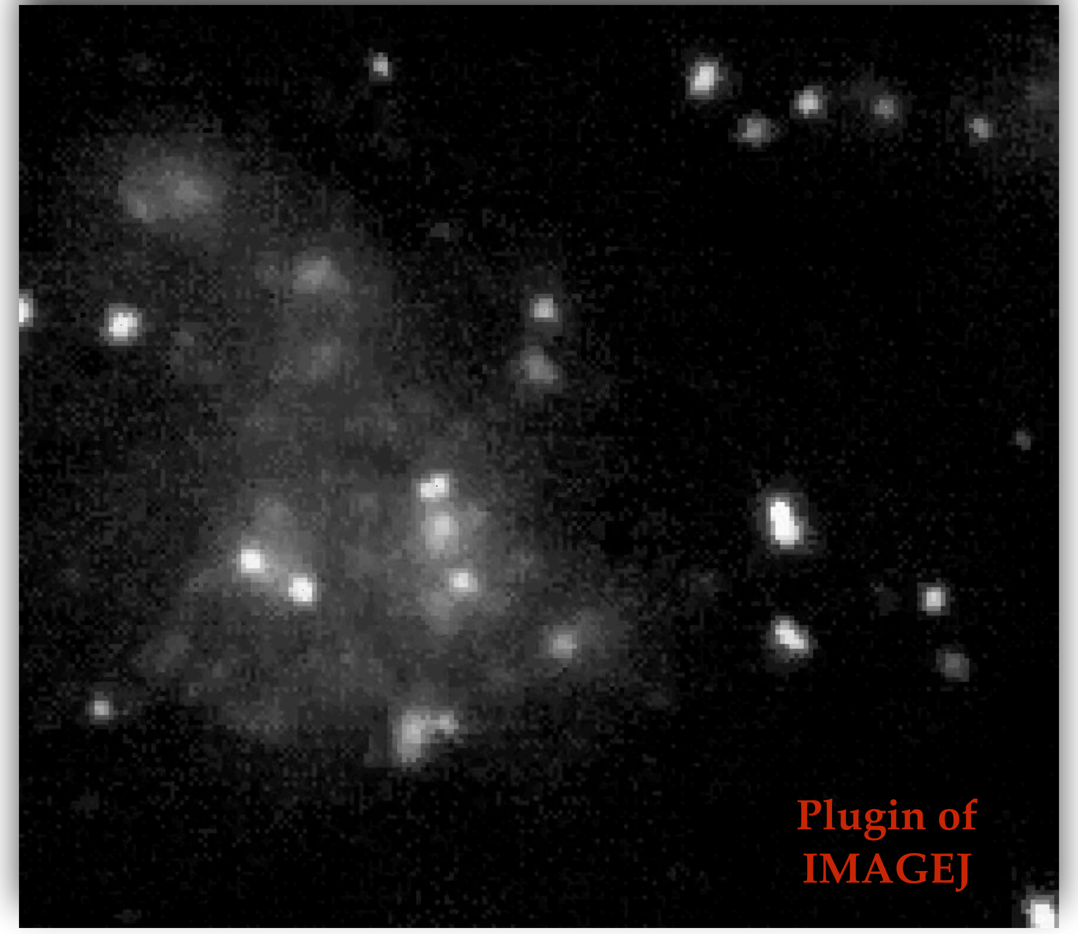
SIMULATIONS USING PARTICLES @ CSE LAB

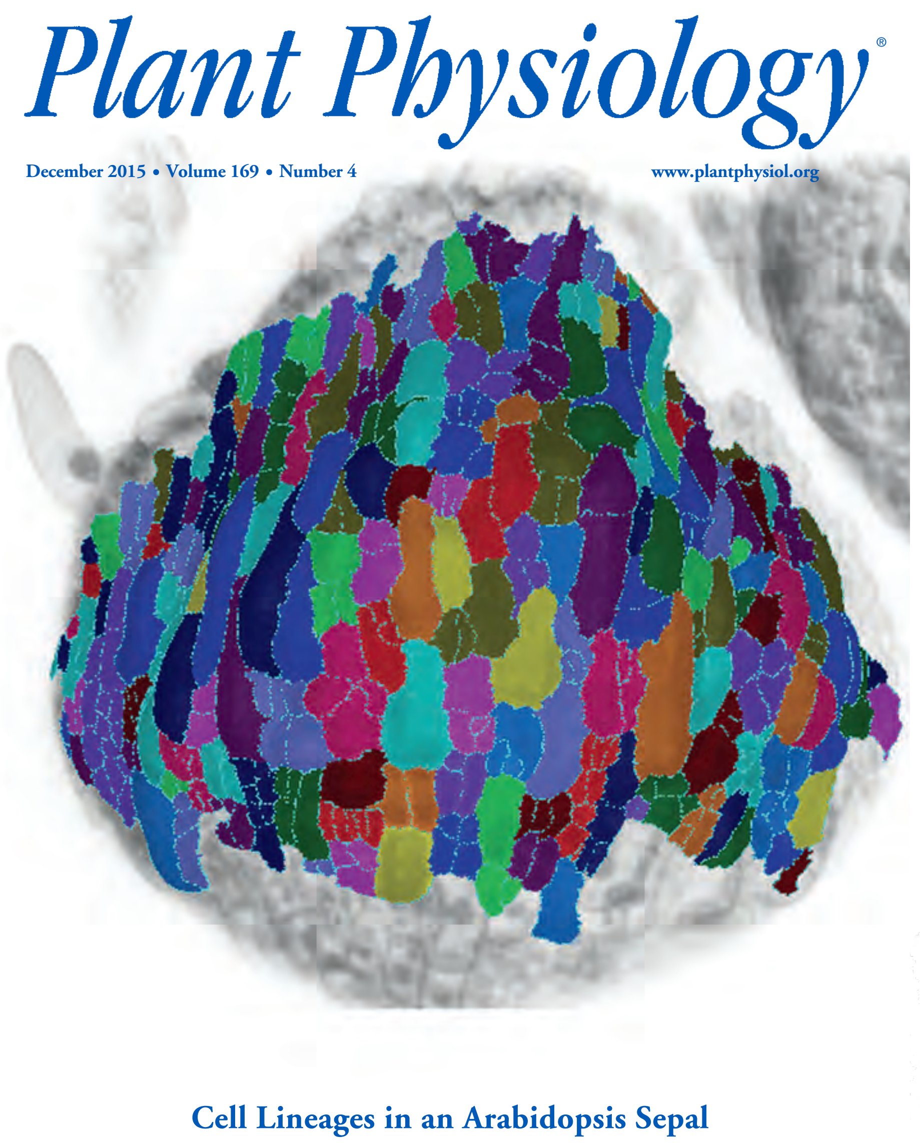
PARTICLES

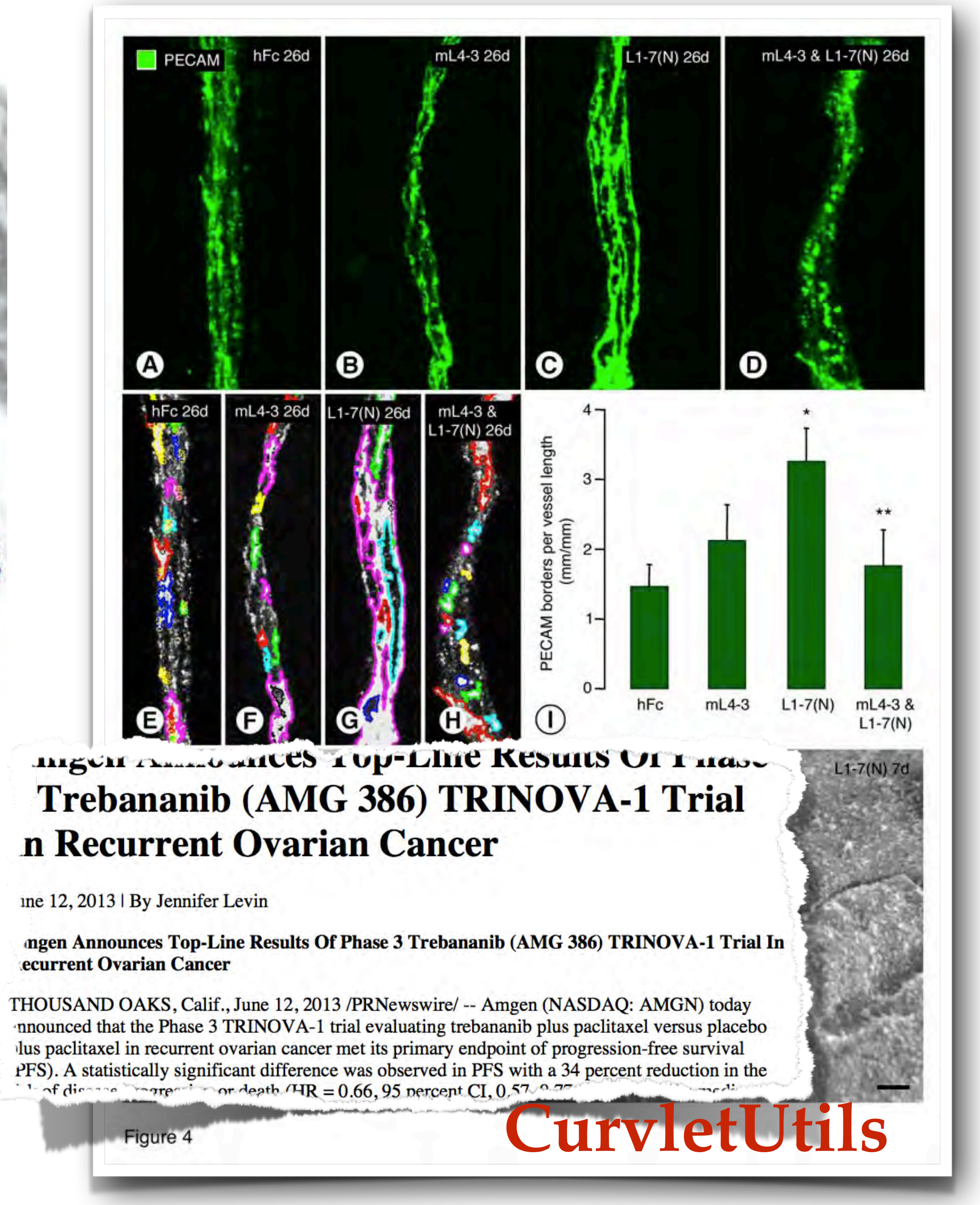
RESEARCH at the CSE Lab

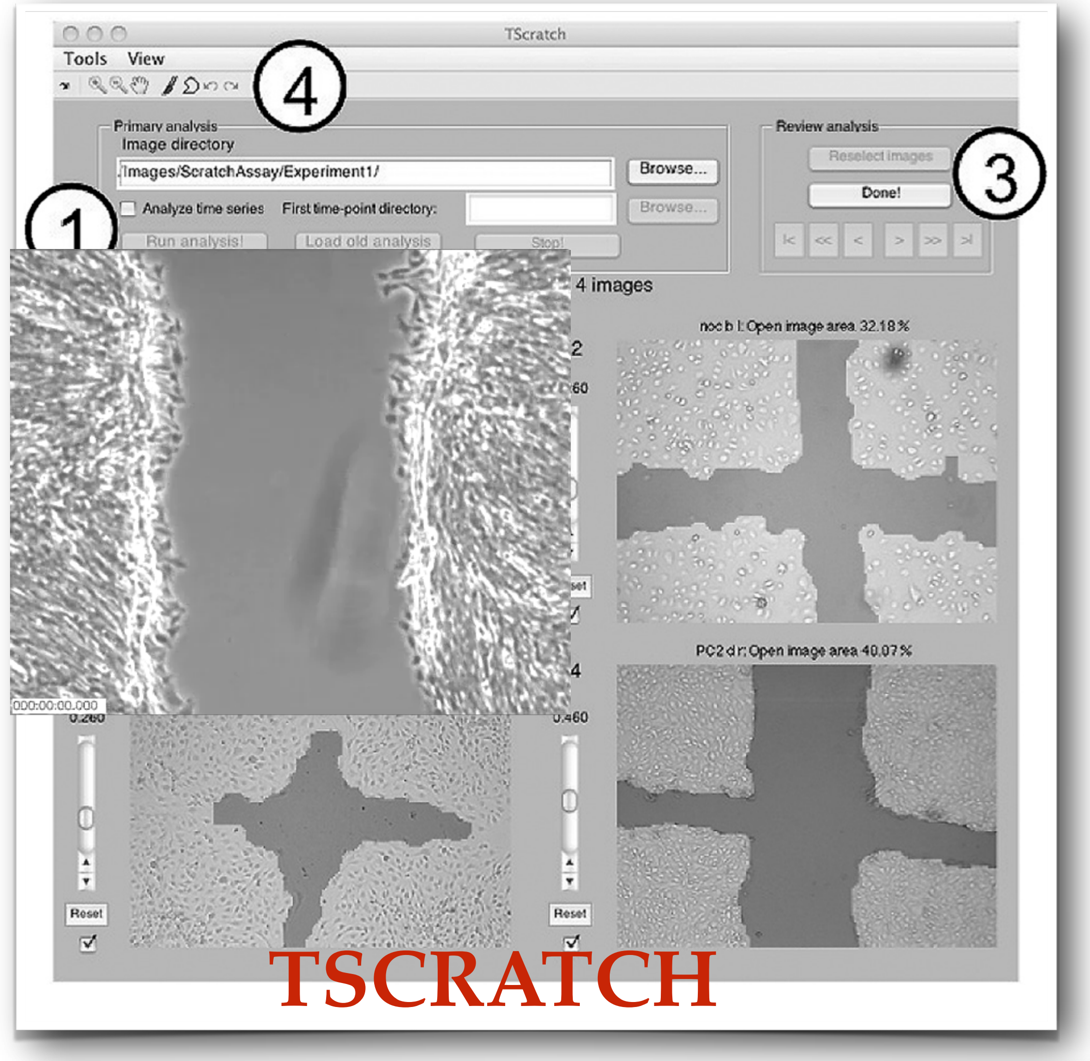


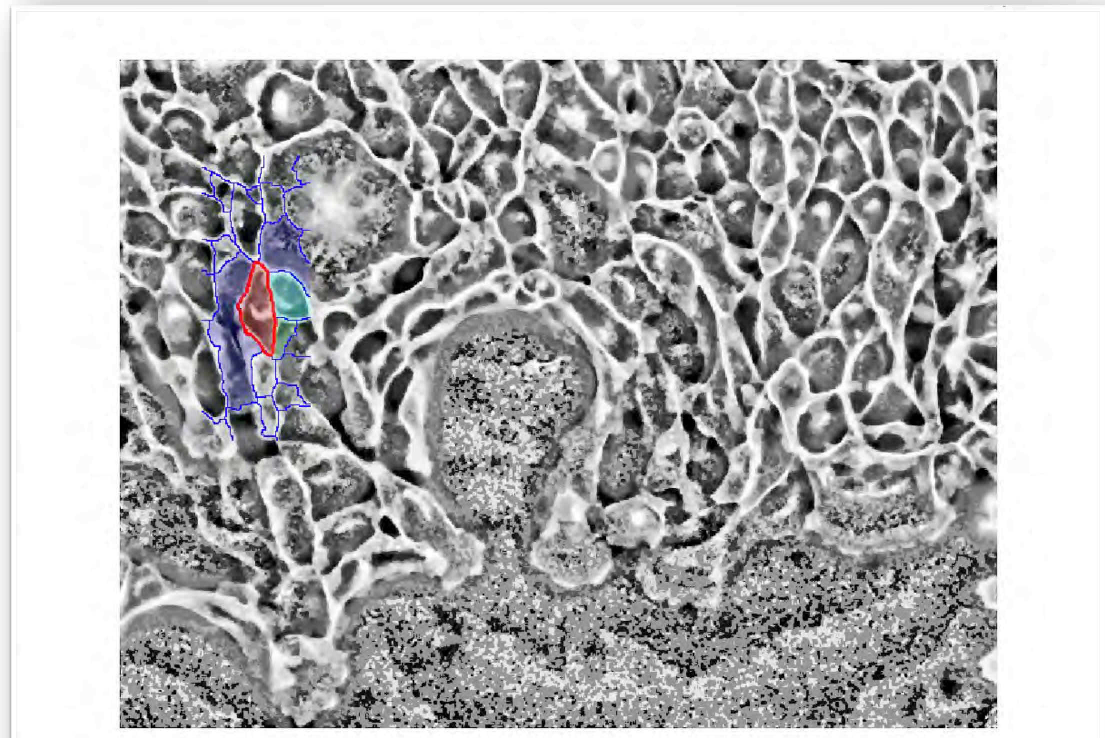
IMAGING



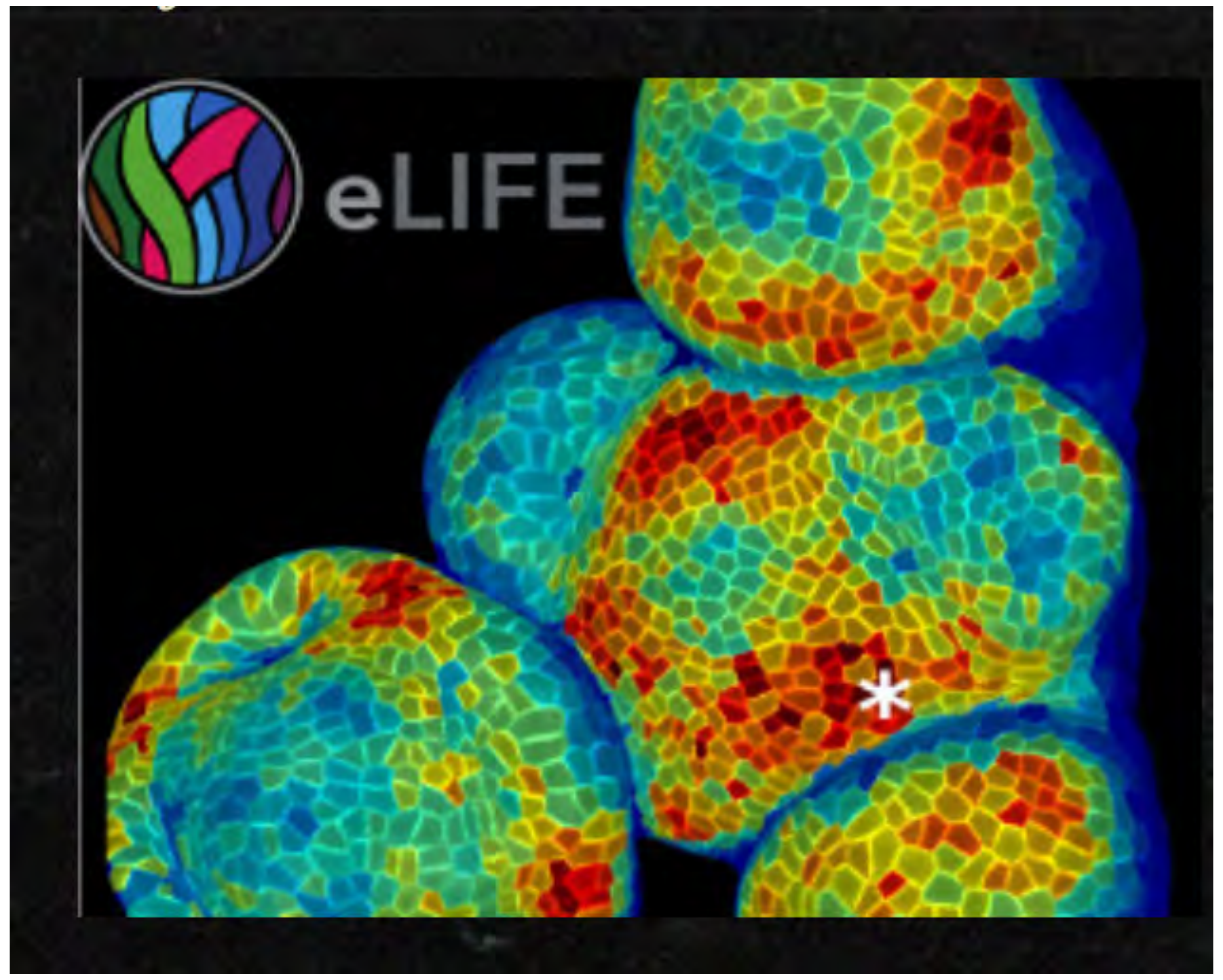






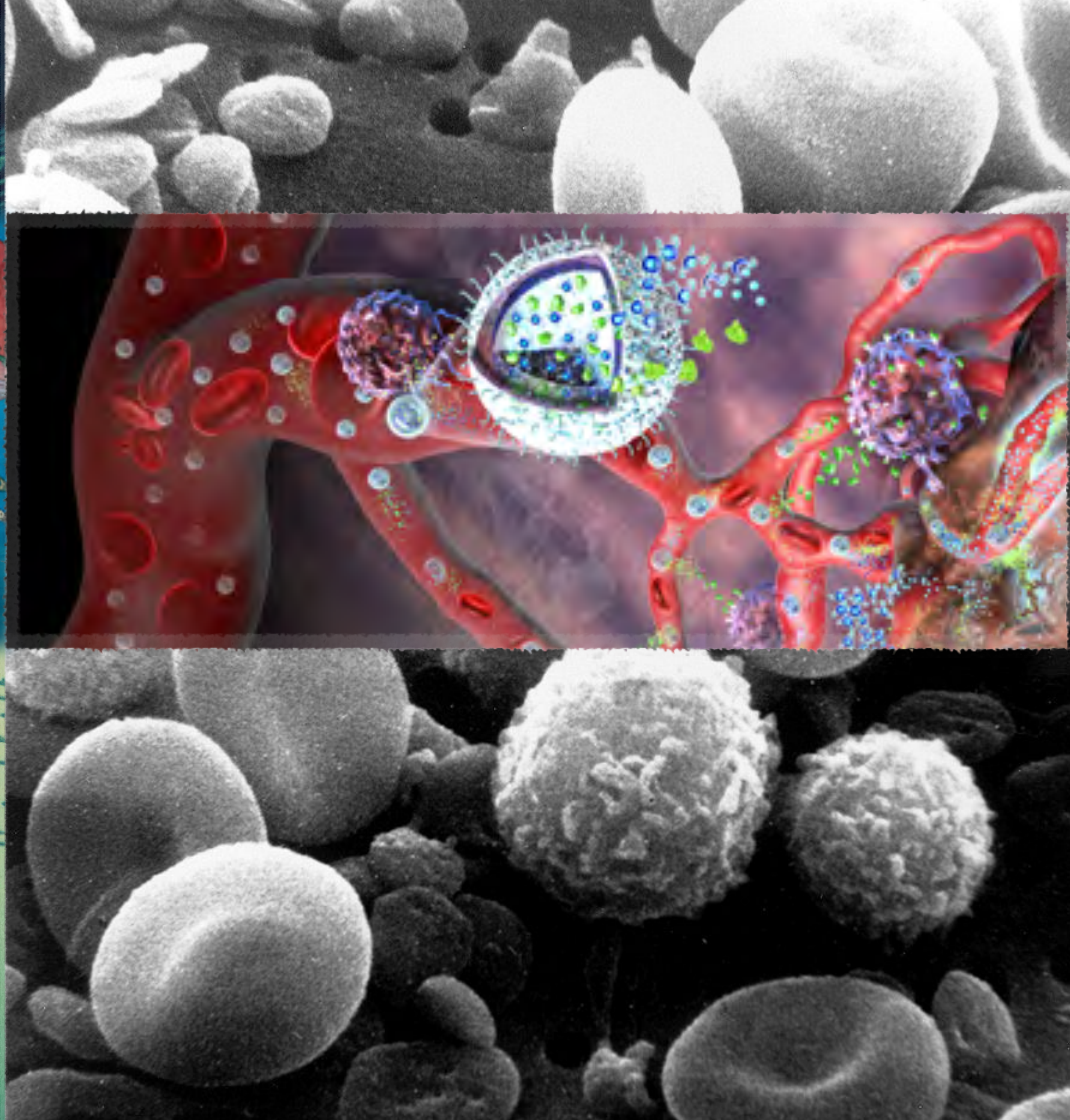


CIV



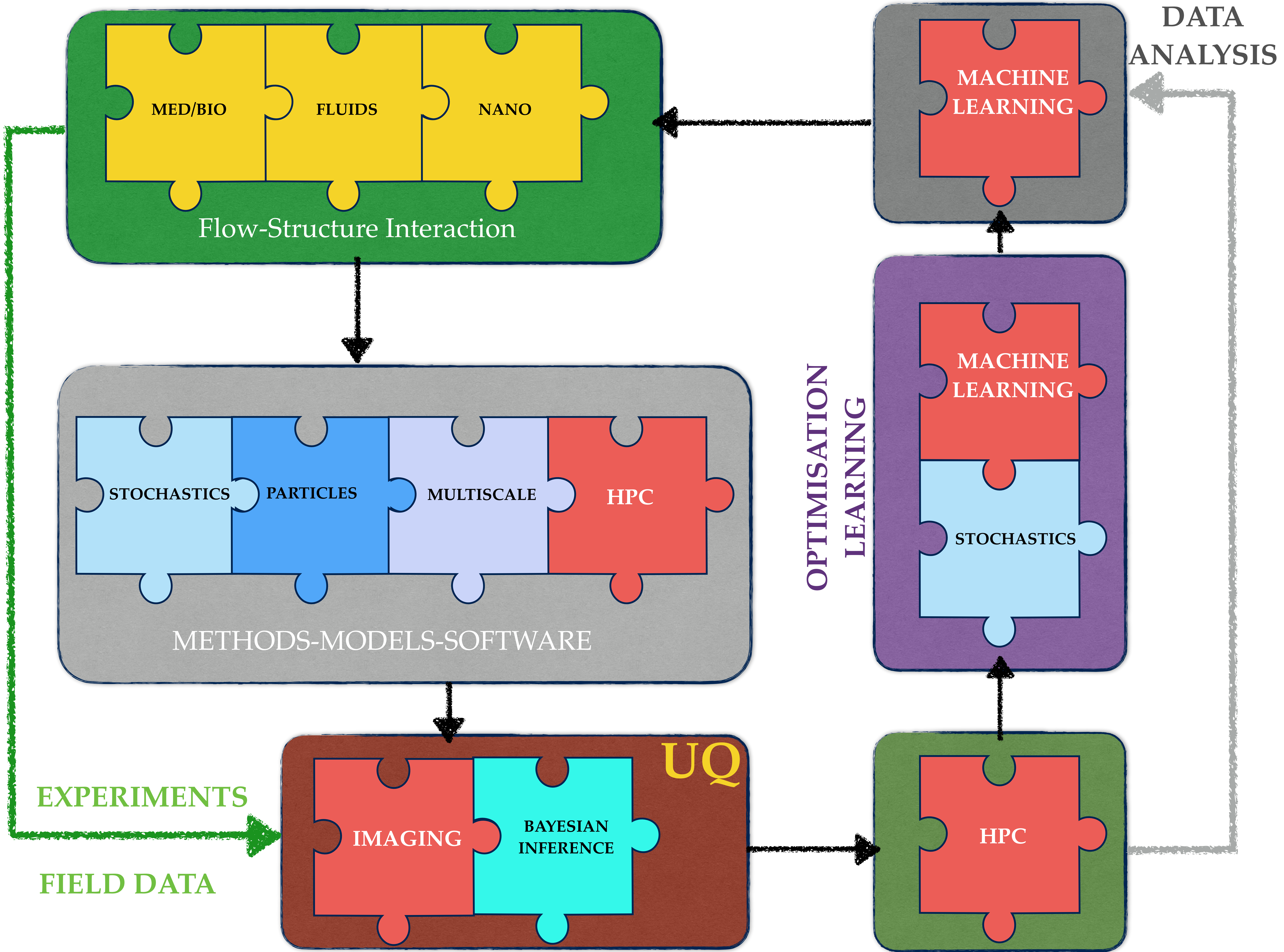
MorphographX

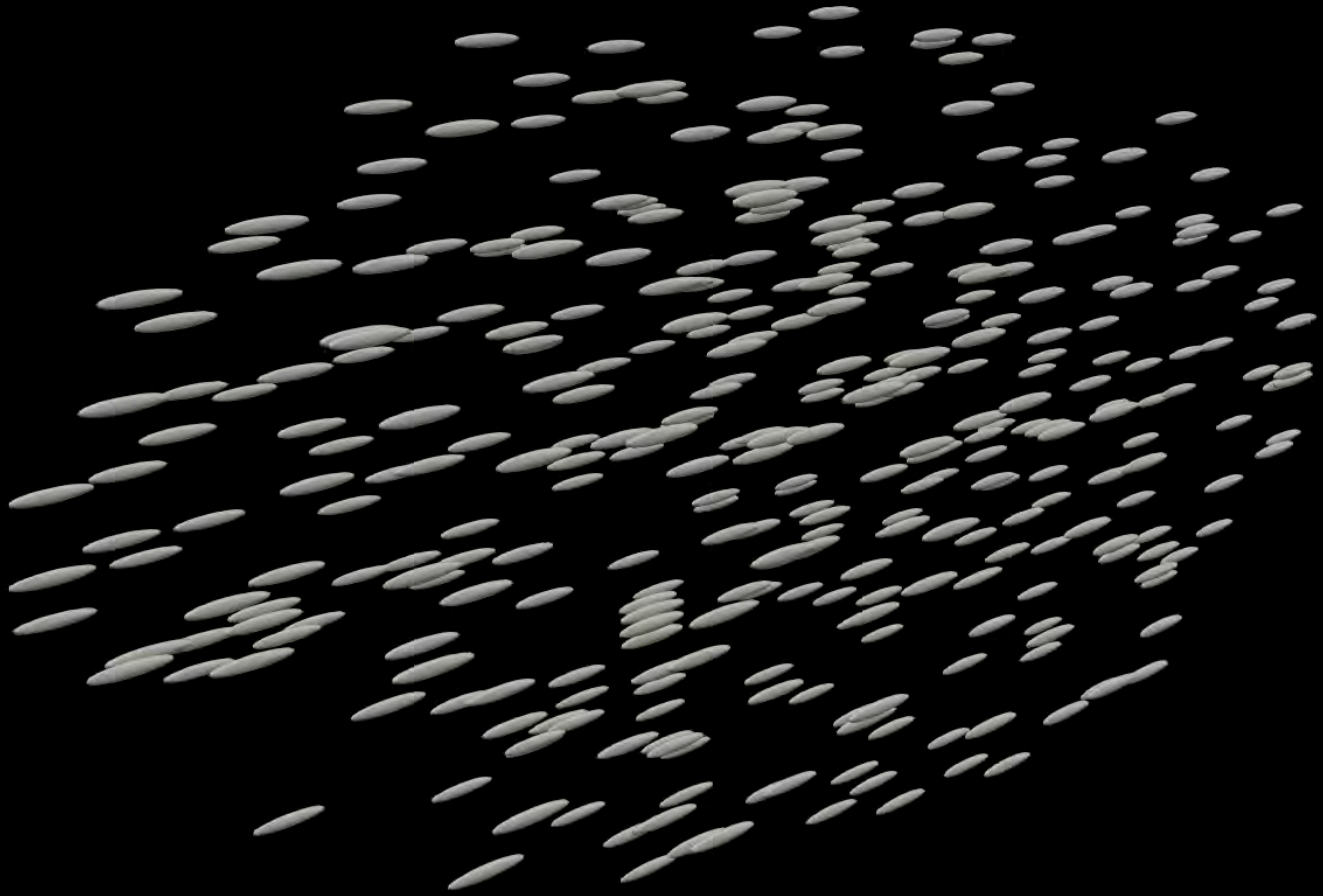

STOCHASTICS
MACHINE LEARNING
U.S. Patent
Jul. 4, 2000
Fig.3A
CREDIT SUISSE



Computing = Joining Boundaries

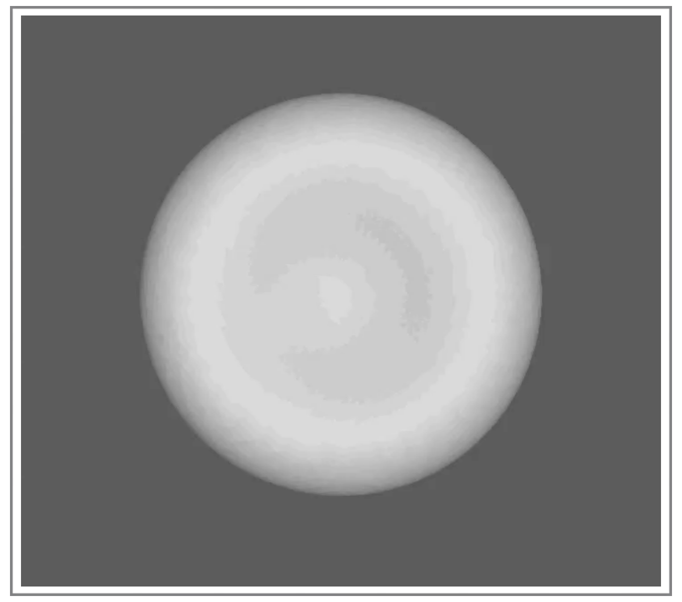
European Research Council

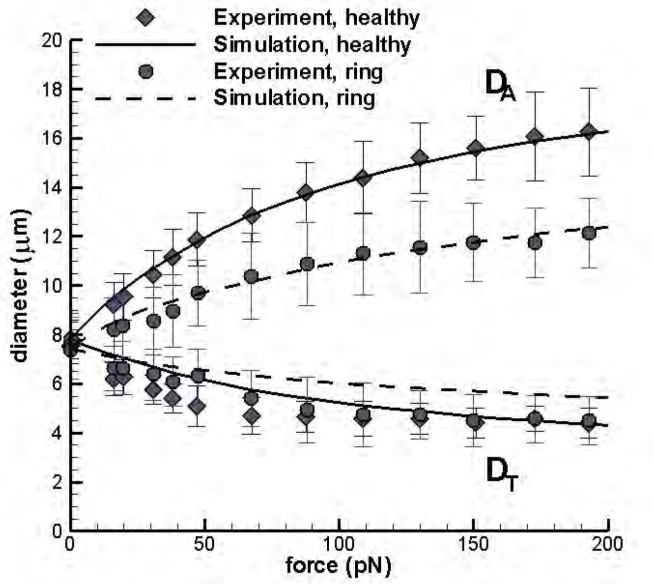

DATA
ANALYSIS
MED/BIO
FLUIDS
NANO
Flow-Structure Interaction
MACHINE
LEARNING
STOCHASTICS
PARTICLES
MULTISCALE
HPC
METHODS-MODELS-SOFTWARE
OPTIMISATION
LEARNING
MACHINE
LEARNING
STOCHASTICS
UQ
EXPERIMENTS
IMAGING
BAYESIAN
INFERENCE
HPC
FIELD DATA



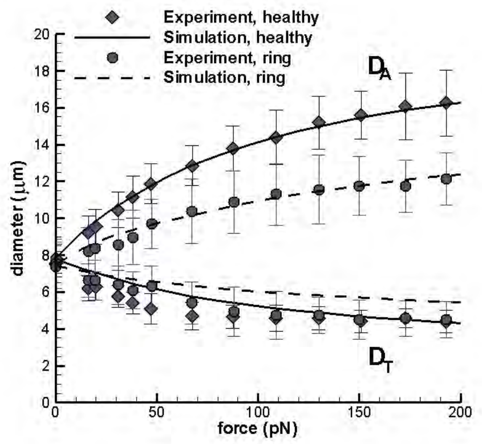
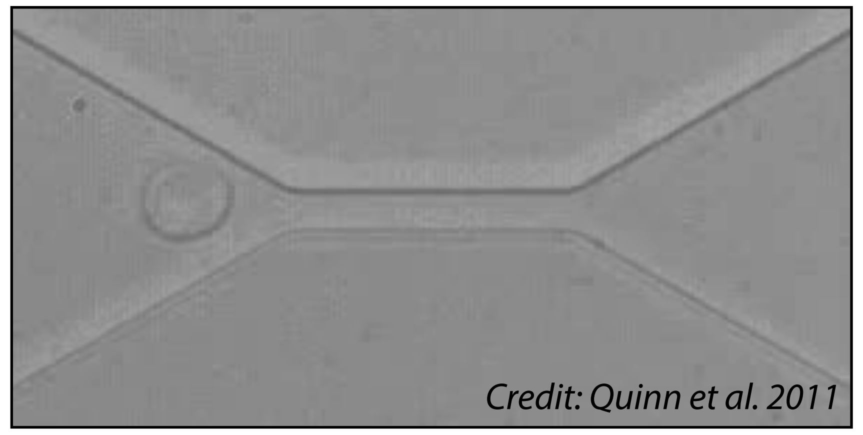
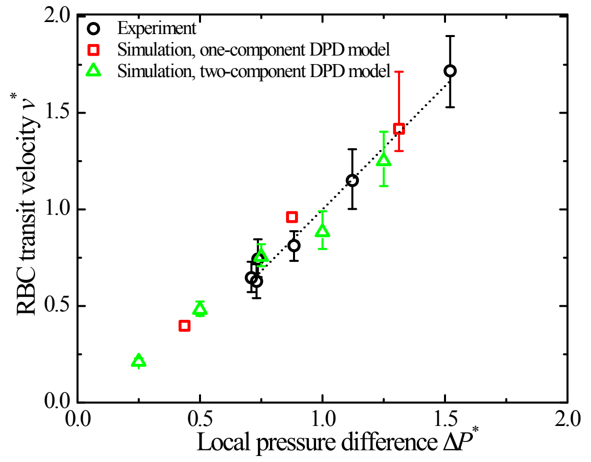
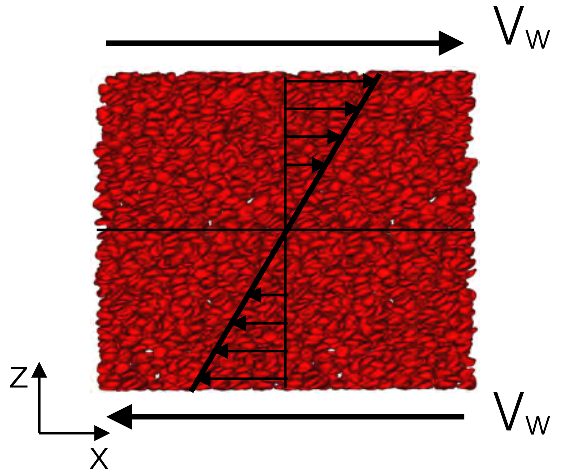
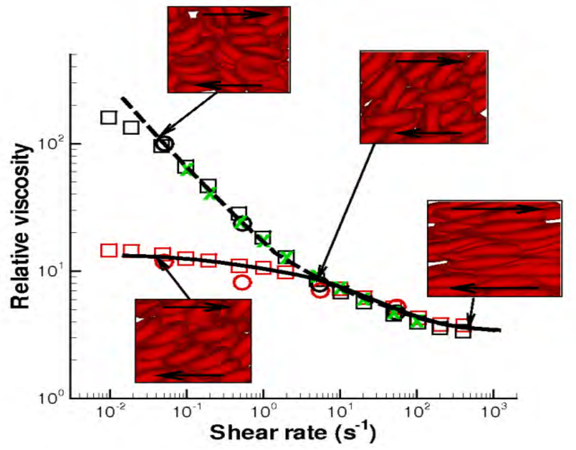
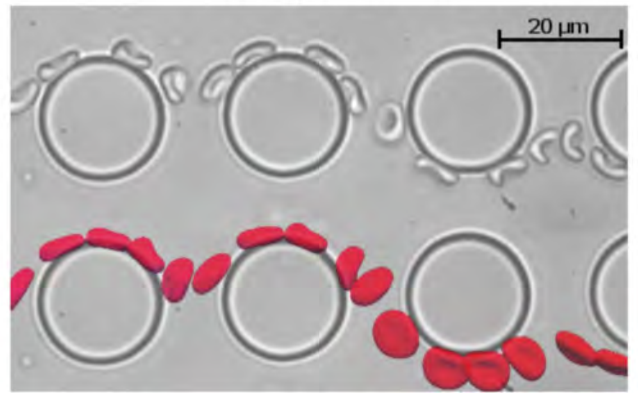
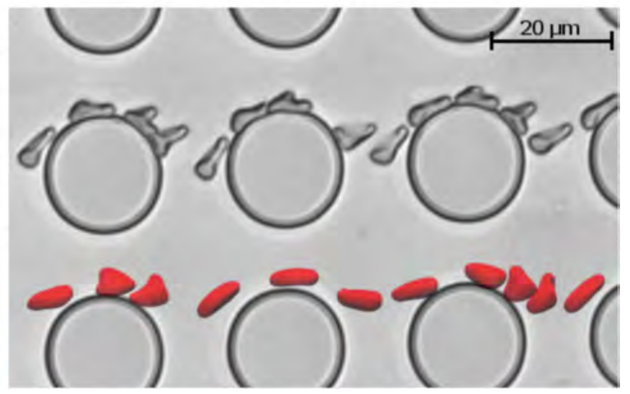
THE in-silico LAB-ON-A-CHIP

VALIDATION




Experiment, healthy
Simulation, healthy
Experiment, ring
Simulation, ring
D_A
D_T
diameter (μm)
force (pN)

DPD validation: Different experiments - different model parameters

TEST CASE	Solvent Model / RBCmodel	Membrane rigidity	Membrane viscosity	Maximum spring extension	Persistence length	viscosity-contrast
stretching (Fedosov et al., 2010)  	DPD / stress-free	4.8E-19 [J]	0.022 [Pa.s]	1.23 e-6 [m]	1.99 e-9 [m]	1
squeeze in micro-channel (Bow et al., 2011)  	DPD / constant spring eq. length	7.5 E-19 [J]	varied to study its effect	3.17 e-7 [m]	1.08 e-7 [m]	1
shear of whole blood (Fedosov et al., 2011)  	DPD / stress-free	3.0 E-19 [J]	0.0144 [Pa.s]	1.3e-7 [m]	1.99 e-9 [m]	1
DLD device (Henry et al., 2016)  	SDPD+a / stress-free	4.8 E-19 [J]	0.022 [Pa.s]	1.23 e-6 [m]	1.99 e-9 [m]	5

ADAPTED FROM: JUDEAH PEARL, CAUSALITY

CLOSING THOUGHTS

Comment

https://doi.org/10.1038/s42254-024-00726-z

On roads less travelled between AI and computational science

Petros Koumoutsakos

Check for updates

Computational science and artificial intelligence have been drivers and benefactors of advances in algorithms and hardware, each in different ways, and originally with different targets. Petros Koumoutsakos argues that the intellectual space between these two fields is home to exciting opportunities for scientific discovery.

'importance sampling'. There are plenty of opportunities for cross-fertilizing exchanges in algorithms and their applications. Similarly, stochastic and gradient optimization methods have been developed across both communities, but recent works on automatic differentiation indicate that the paths are intersecting again. The emergence and homogenization properties of foundational models that are gaining ground in AI also have counterparts in CoS where emergence is often the outcome of nonlinear differential equations, whereas the concept of homogenization can be recognized for example in particle simulations of phenomena ranging from atoms to galaxies[3]. At the same time the paths of scientific inquiry in AI and CoS may diverge, but I argue that repeated intersection can be exciting. There are many problems where

Nature Reviews Physics, 2024


SCIENTIFIC COMPUTING
Mathematics
Exactness
Understanding
ALLOYS
ARTIFICIAL INTELLIGENCE
Architectures
Statistics
Goals

PART II

LECTURES

- **LECTURE 1:**
  - What is Artificial Intelligence? How it relates to Computing ?
  - How are these relevant to modeling, understanding, optimizing complex systems?

- **LECTURE 2:**
  - (Machine) Learning of Effective Dynamics of Complex Systems.
  - Reinforcement Learning for Controlling Complex Systems.

- **LECTURE 3**
  - Learning to solve PDEs without Machine Learning
  - Learning to solve PDEs with Reinforcement Learning Machine

- **LECTURE 4**
  - Uncertainty Quantification & Stochastic Optimization
  - Optimal Sensor Placement

How to Fly with a Broken Wing

NASA Photo EC89 232-1

An Israeli Air Force F-15 involved in a midair collision during a training mission in 1983. In spite of losing virtually the entire starboard wing, the pilot successfully landed the jet.

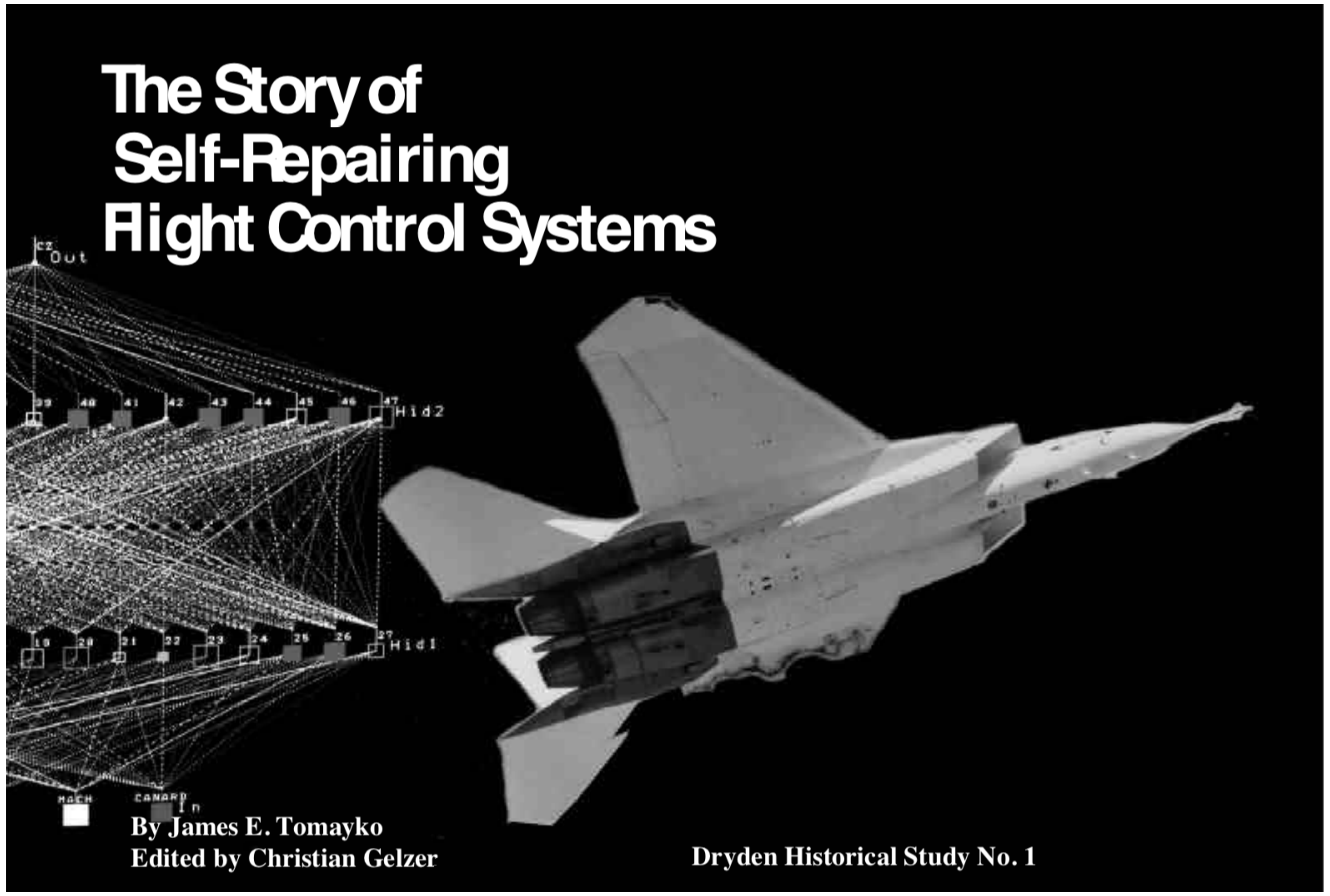
Zivi Nedivi

NASA Technical Memorandum 112198

Direct Adaptive Aircraft Control Using Dynamic Cell Structure Neural Networks

Charles C. Jorgensen, Ames Research Center, Moffett Field, California

May 1997



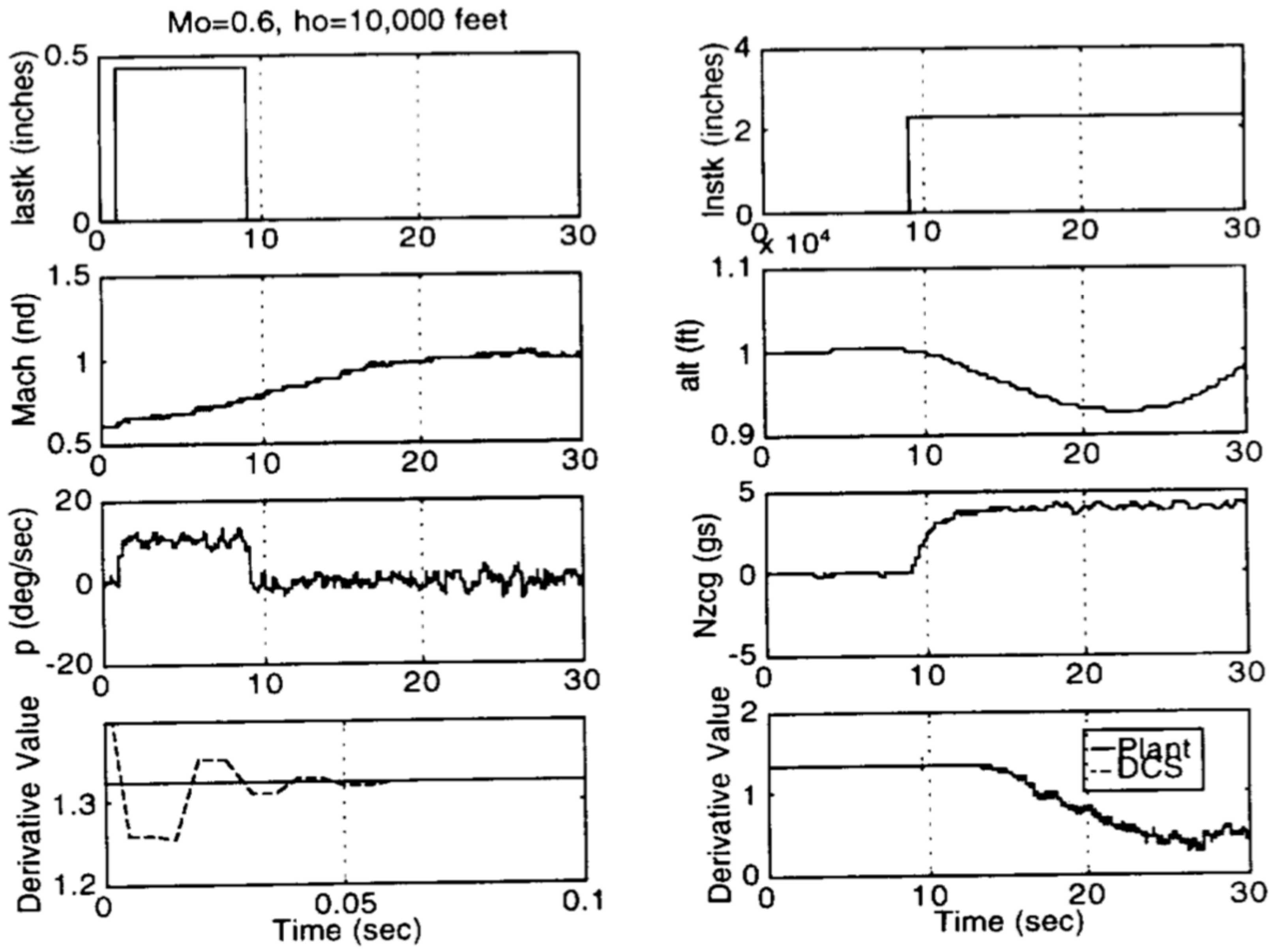


Figure 5(c). Learning with turbulence.

NASA Technical Memorandum 112197

Neural Network Prediction of New Aircraft Design Coefficients

Magnus Nørgaard, Institute of Automation, Technical University of Denmark
Charles C. Jorgensen, Ames Research Center, Moffett Field, California
James C. Ross, Ames Research Center, Moffett Field, California

May 1997

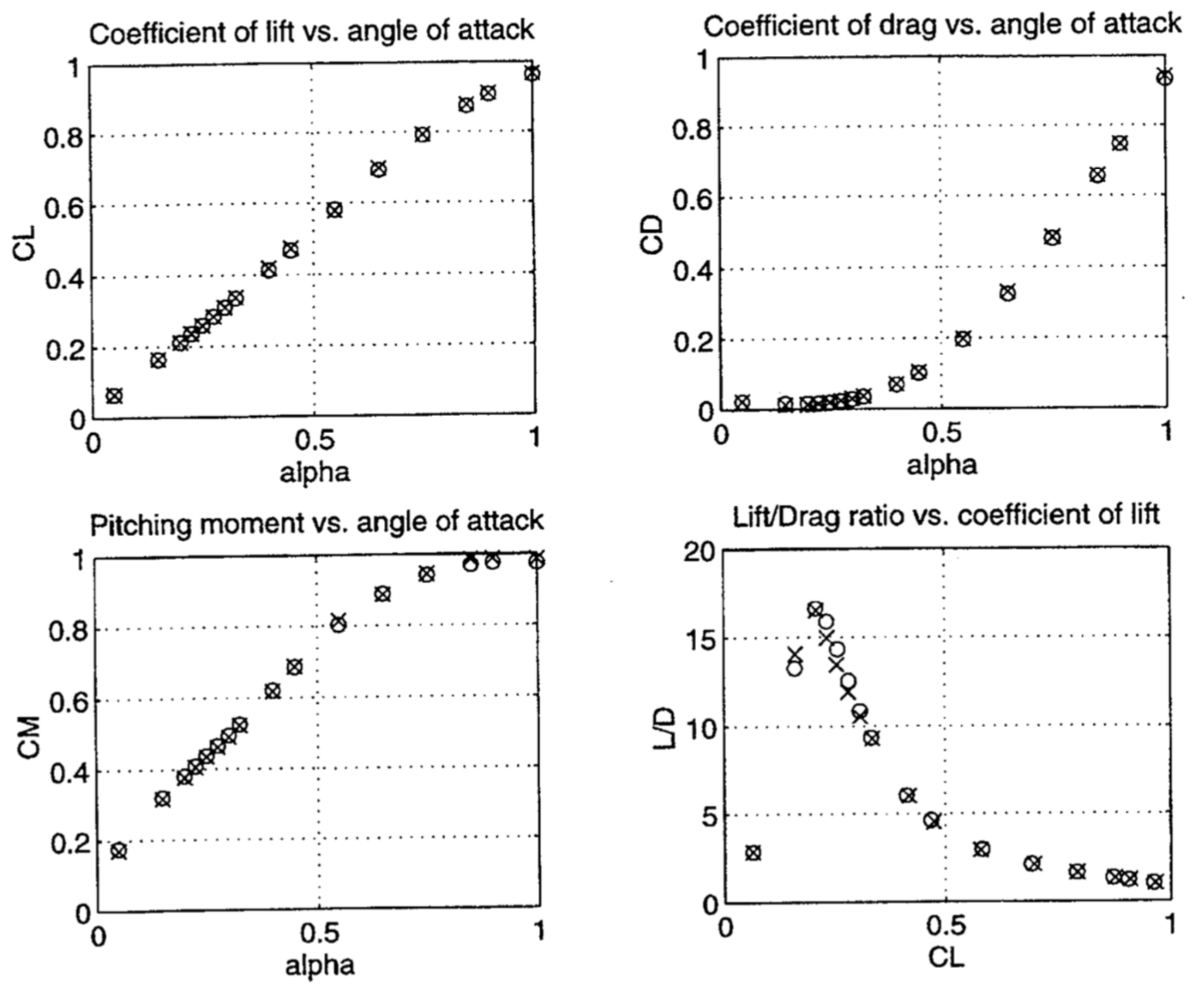
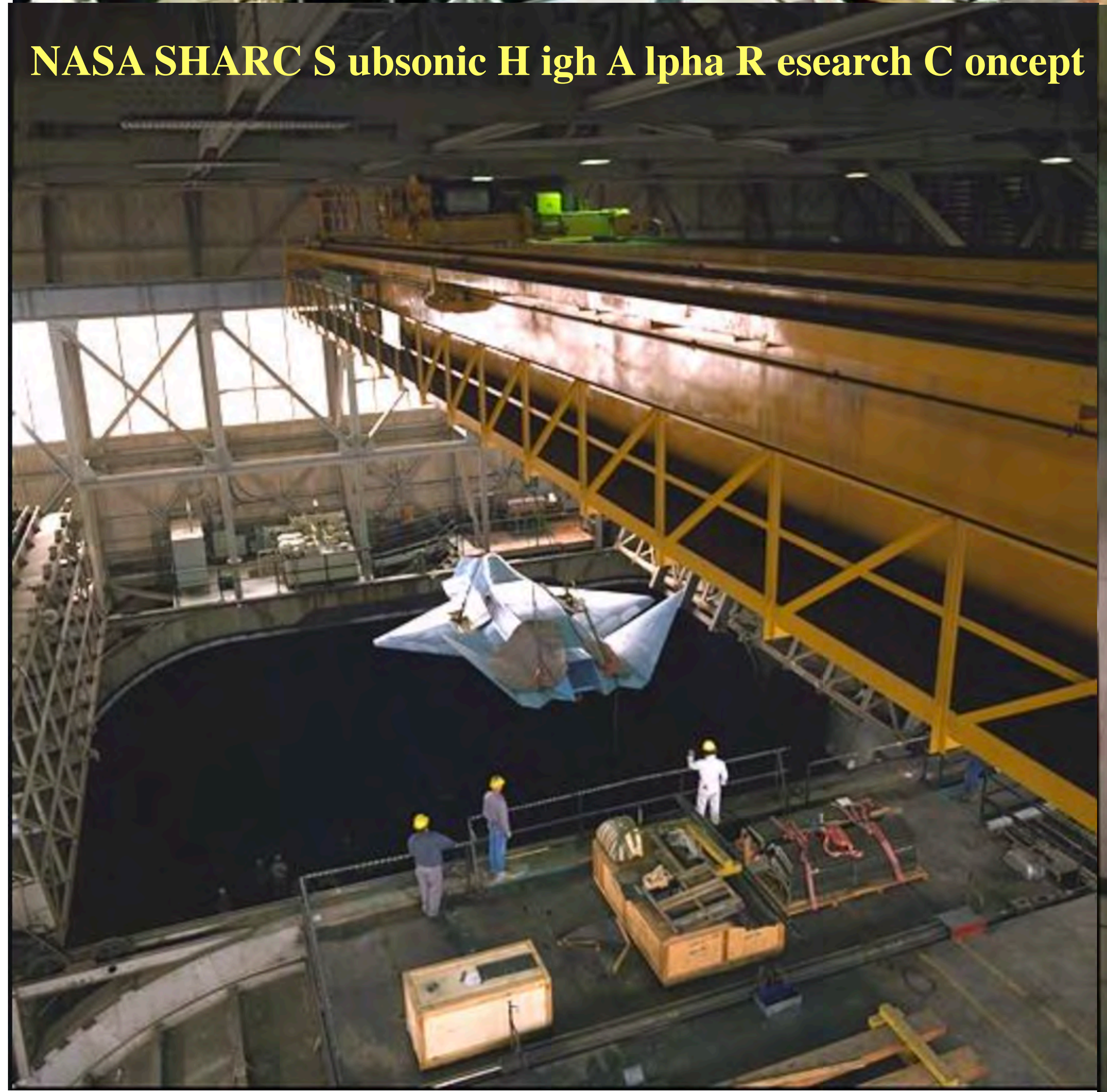


Figure 4. Comparison of test data and network predictions for LE=0.50 and TE=0.0



MACHINE LEARNING

Capabilities for Pattern Recognition

Dimensionality Reduction: **POD/PCA as NN**

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

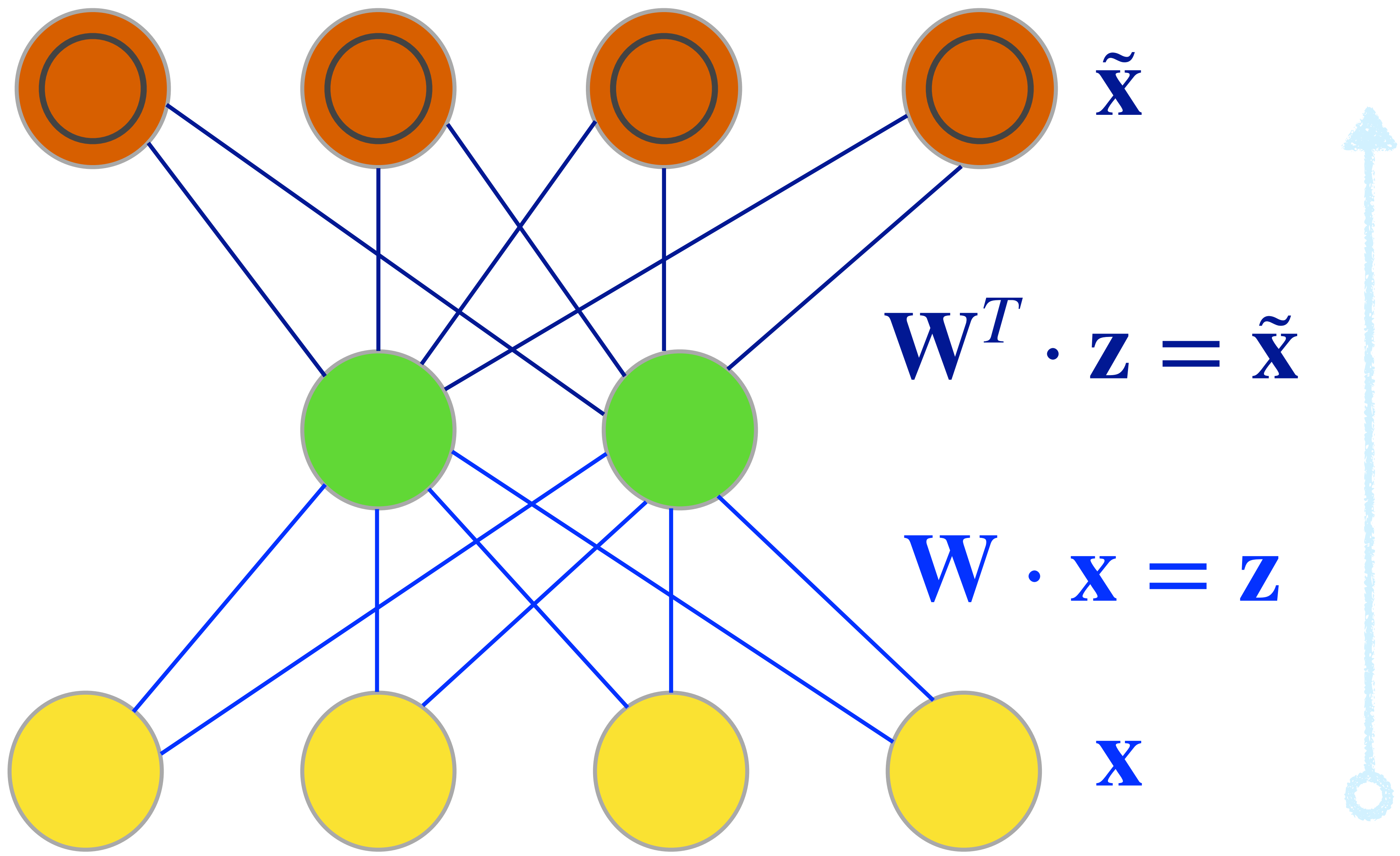
$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T$$

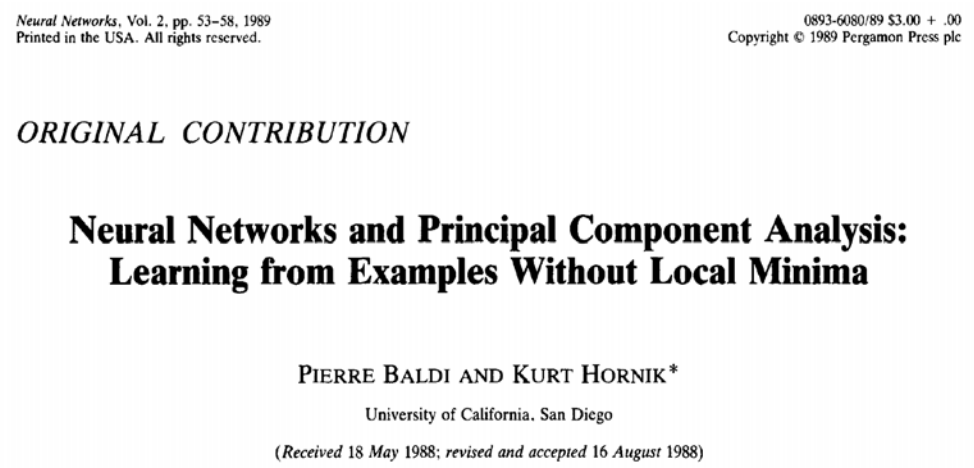
$$\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

`retain M < D eigenvectors`

$$E = ||\tilde{\mathbf{x}} - \mathbf{x}||^2$$
$$= ||\mathbf{W}^{\mathrm{T}}\mathbf{W}\cdot\mathbf{x} - \mathbf{x}||^2$$

Find $\mathbf{W}$ by minimizing E




Neural Networks, Vol. 2, pp. 53–58, 1989
Printed in the USA. All rights reserved.
0893-6080/89 $3.00 + .00
Copyright © 1989 Pergamon Press plc

ORIGINAL CONTRIBUTION

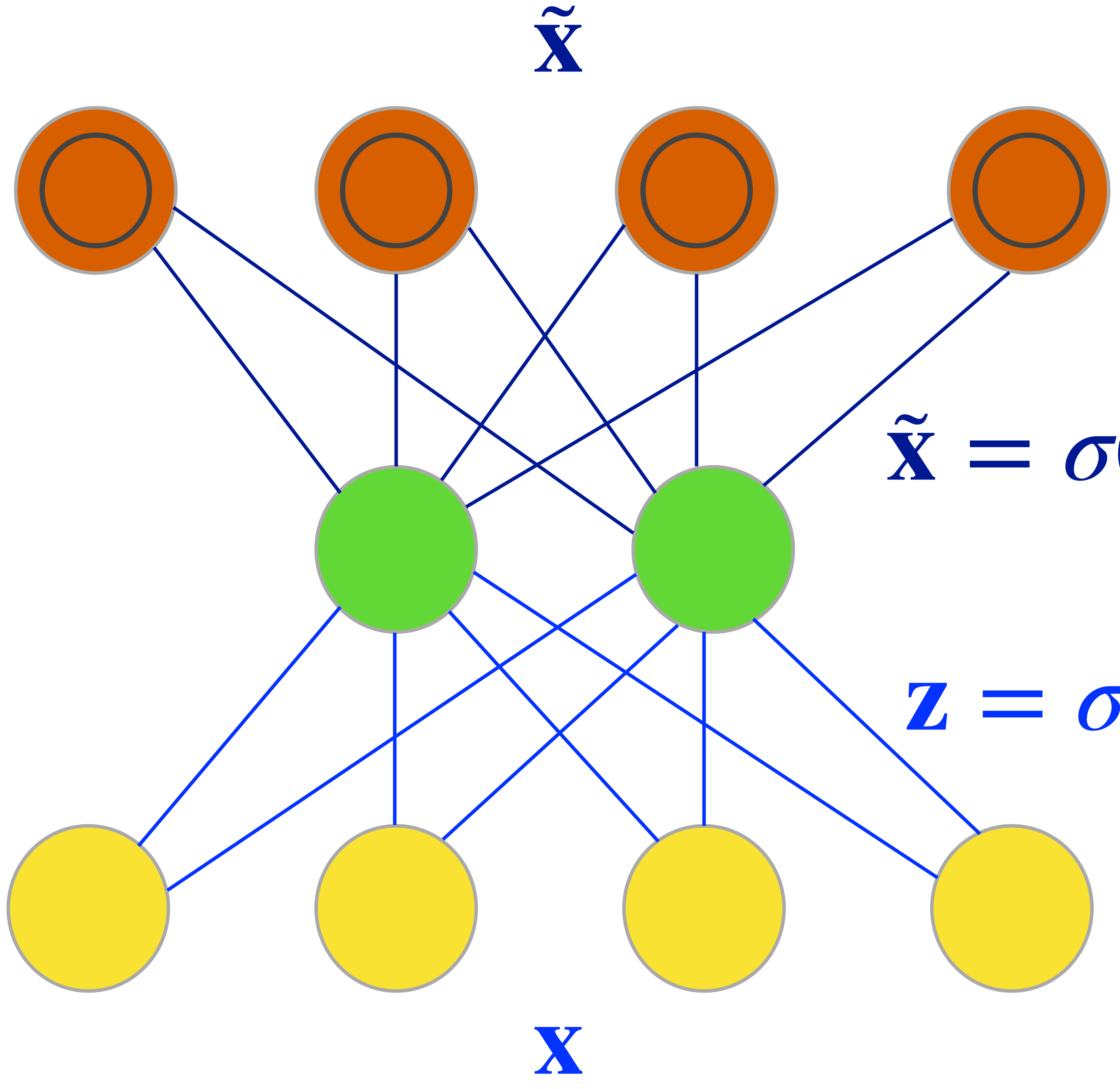
Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima

Pierre Baldi and Kurt Hornik*

University of California, San Diego

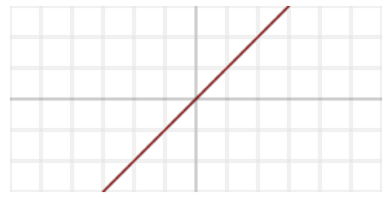
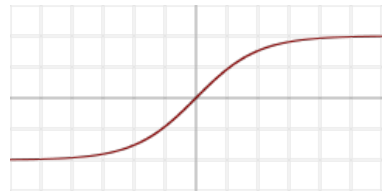
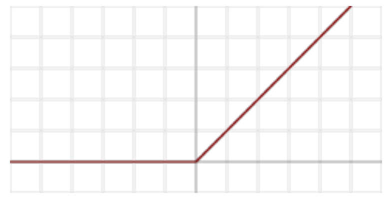
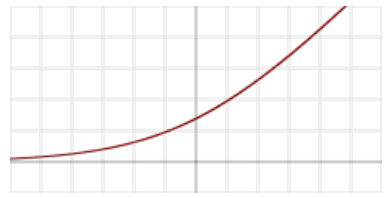
(*Received* 18 *May* 1988; *revised and accepted* 16 *August* 1988)

Non-Linear PCA - Autoencoders



$$\tilde{\mathbf{x}} = \sigma(\mathbf{W}_2 \cdot \mathbf{z} + b_2)$$

$$\mathbf{z} = \sigma(\mathbf{W}_1 \cdot \mathbf{x} + b_1)$$

activation function σ		derivative
$\sigma(x) = x$		$\sigma'(x) = 1$
$\sigma(x) = \dfrac{2}{1+e^{-2x}} - 1$		$\sigma'(x) = 1 - f(x))^2$
$\sigma(x) = max(0, x)$		$\sigma'(x) = max(0, 1)$
$\sigma(x) = ln(1 + e^x)$		$\sigma'(x) = \dfrac{1}{1+e^{-x}}$

$$E = ||\tilde{\mathbf{x}} - \mathbf{x}||^2$$

- learns a function with target values equal to the input
- linear auto-encoder “equivalent” to PCA/POD

Journal of Computational Physics **182,** 1–26 (2002)
doi:10.1006/jcph.2002.7146

Neural Network Modeling for Near Wall Turbulent Flow

Michele Milano[1] and Petros Koumoutsakos[2]

Institute of Computational Sciences, ETH Zentrum, CH-8092 Zürich, Switzerland
E-mail: milano@inf.ethz.ch, petros@inf.ethz.ch

Received May 23, 2001; revised January 8, 2002

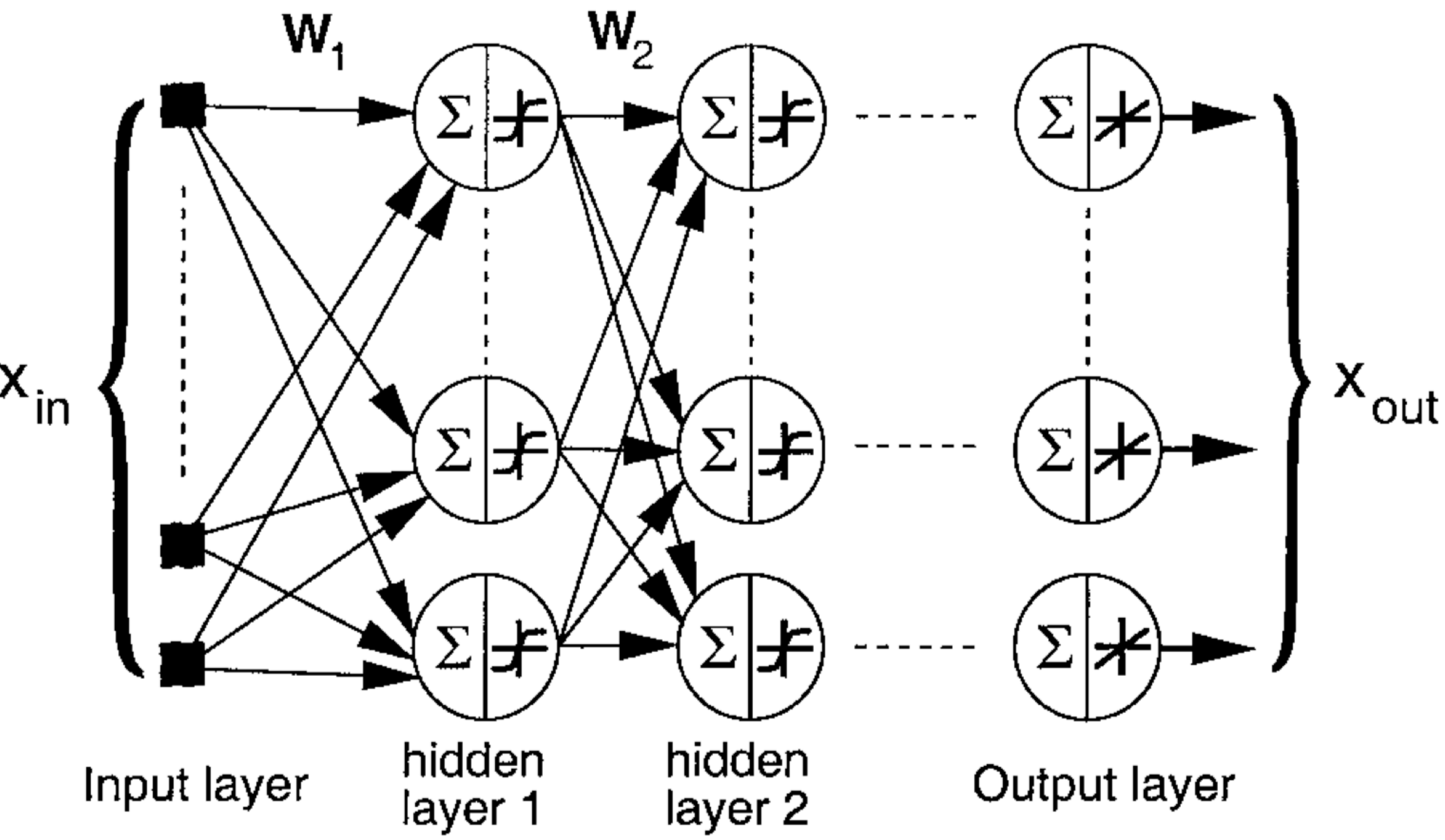
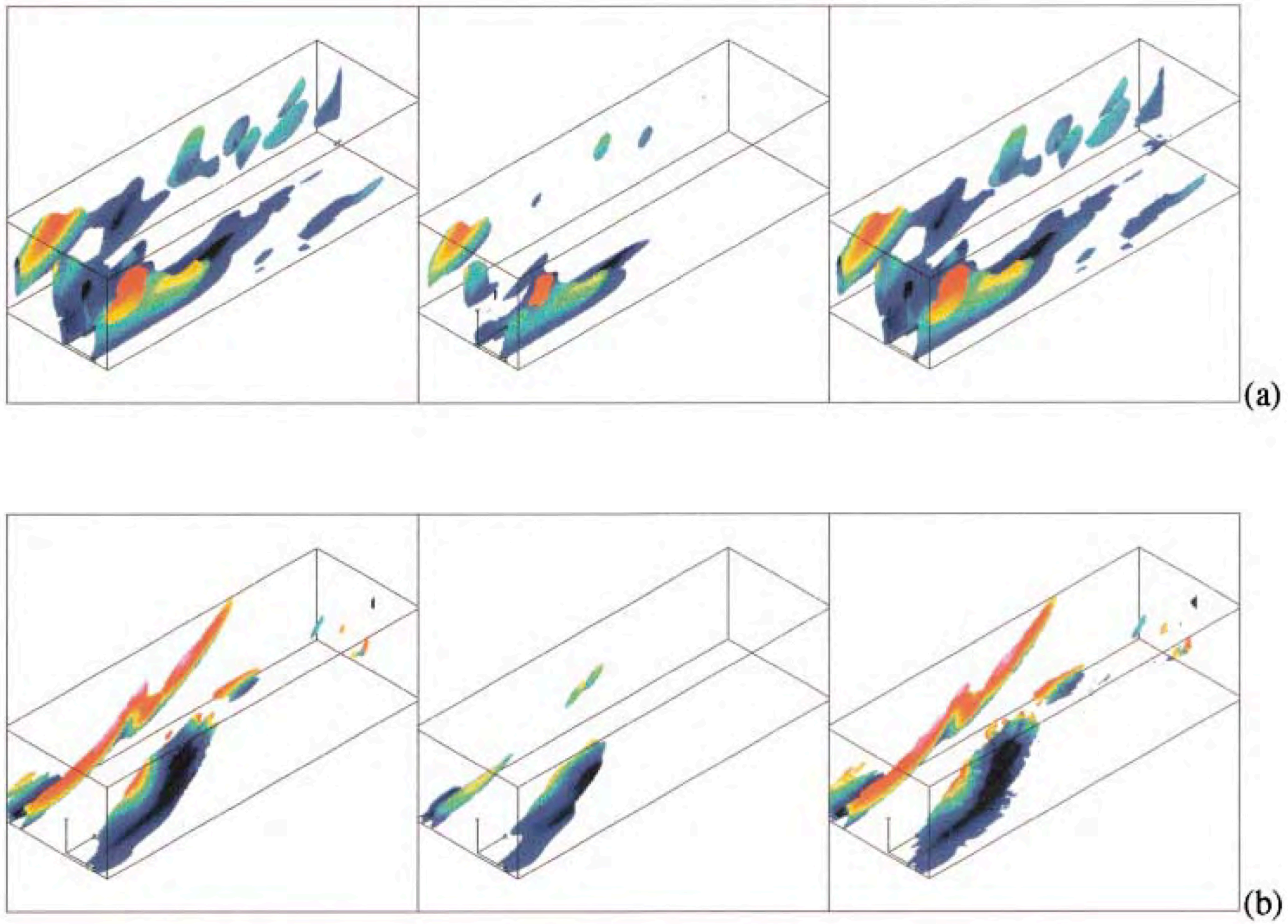
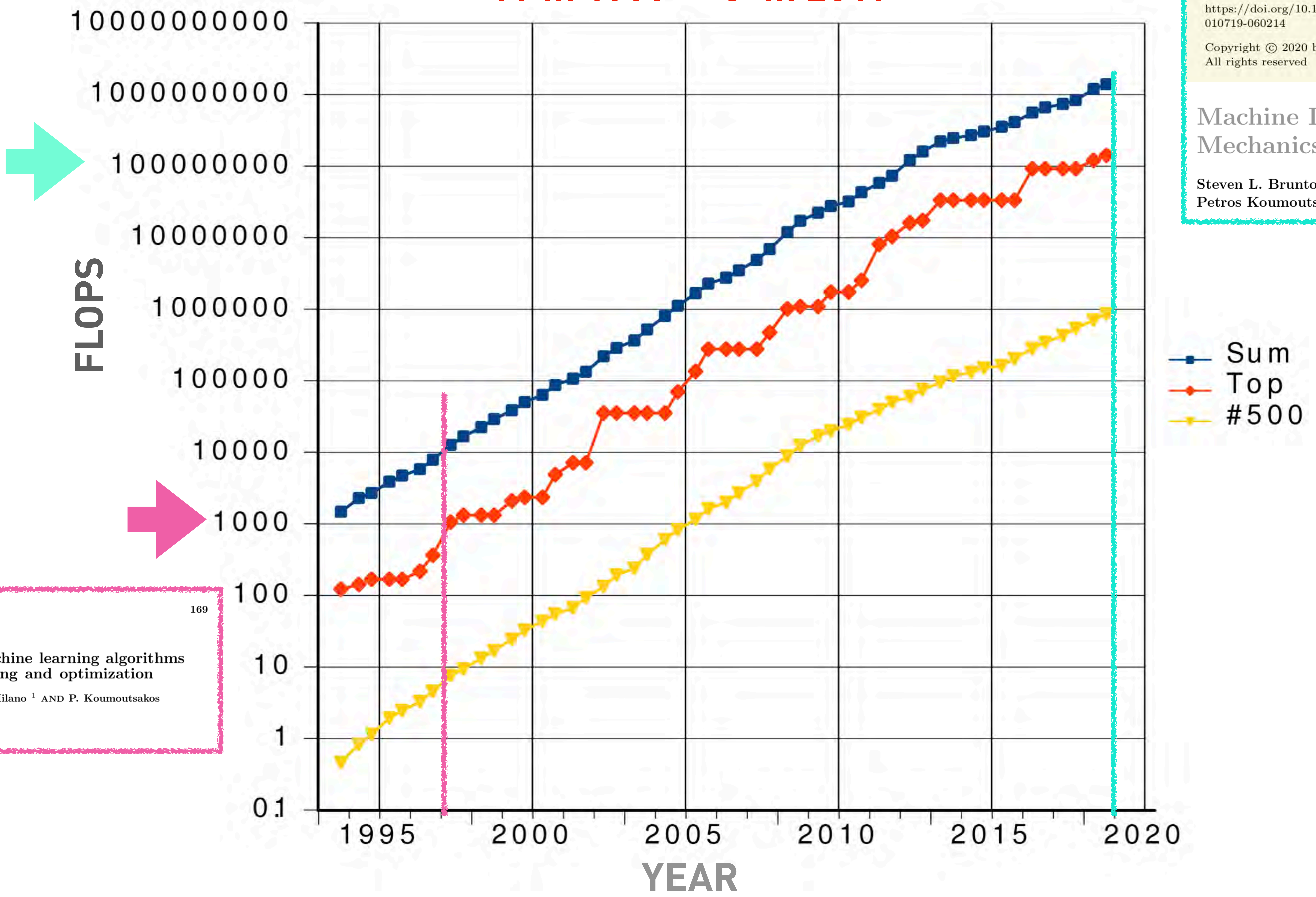


FIG. 1. Layer representation of a nonlinear neural network structure.

ORIGINAL POD NN



1Y in 1997 ~ 3' in 2019



Annu. Rev. Fluid Mech. 2020. 52:1–31

https://doi.org/10.1146/annurev-fluid-010719-060214

Machine Learning for Fluid Mechanics

Steven L. Brunton,[1] Bernd R. Noack,[2] and Petros Koumoutsakos[3,4]

Center for Turbulence Research
Annual Research Briefs 1999 169

Application of machine learning algorithms to flow modeling and optimization

By S. Müller [1], M. Milano [1] AND P. Koumoutsakos

Deep learning’s Big Bang moment.

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

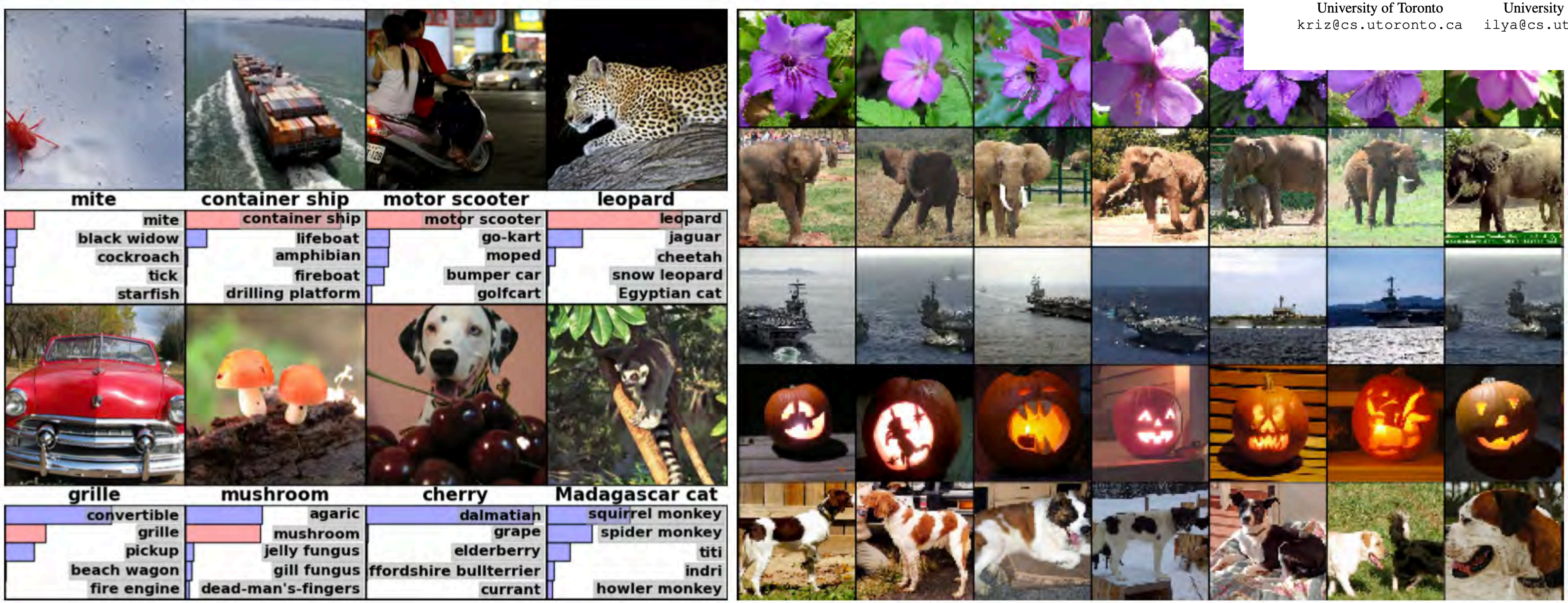
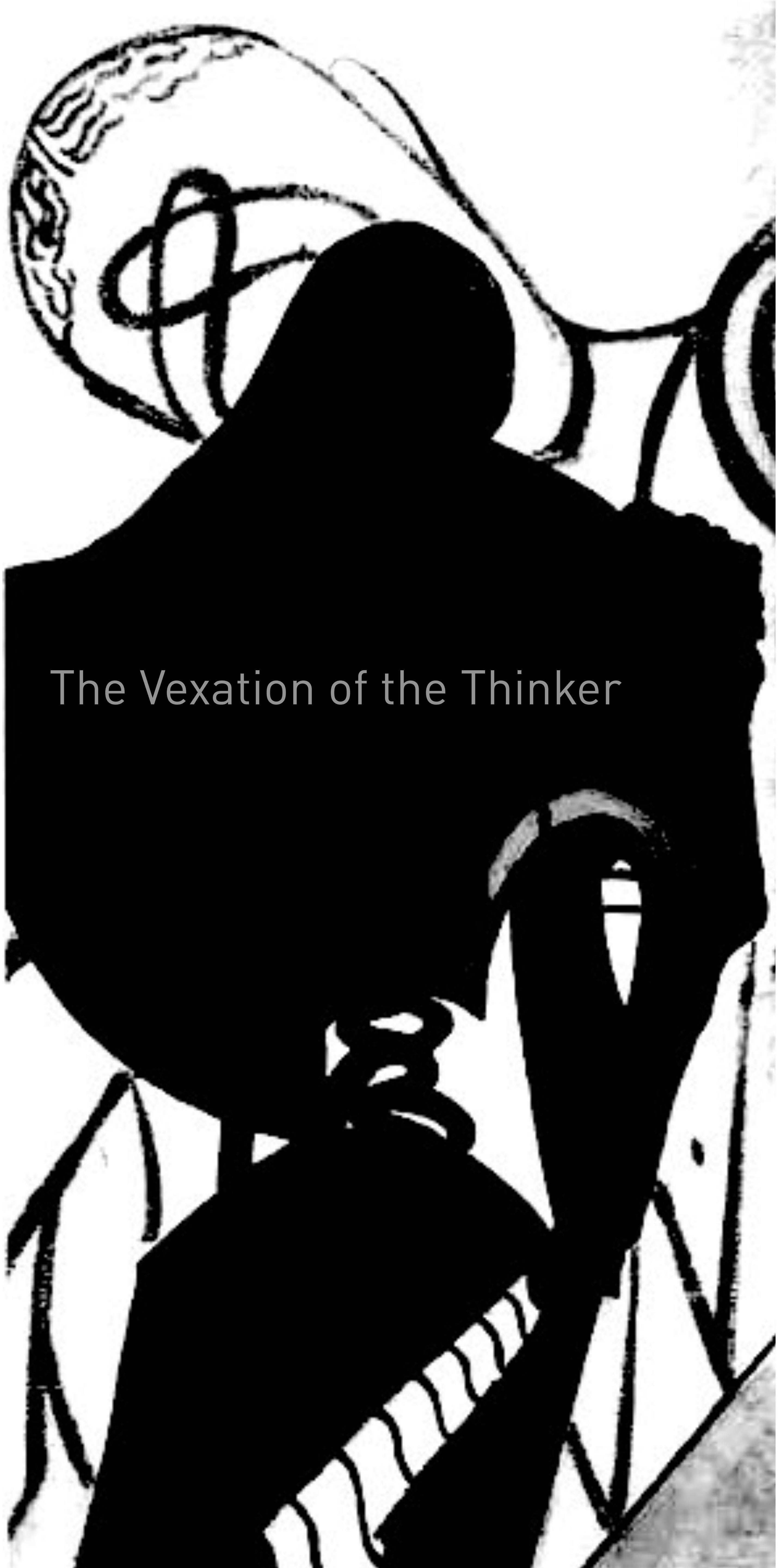


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

Patterns



Machine Learning: Success for Pattern Recognition

Patterns can exist also in Dynamics -
Latent spaces of Dynamical Systems?

Can Machine Learning recognise them?

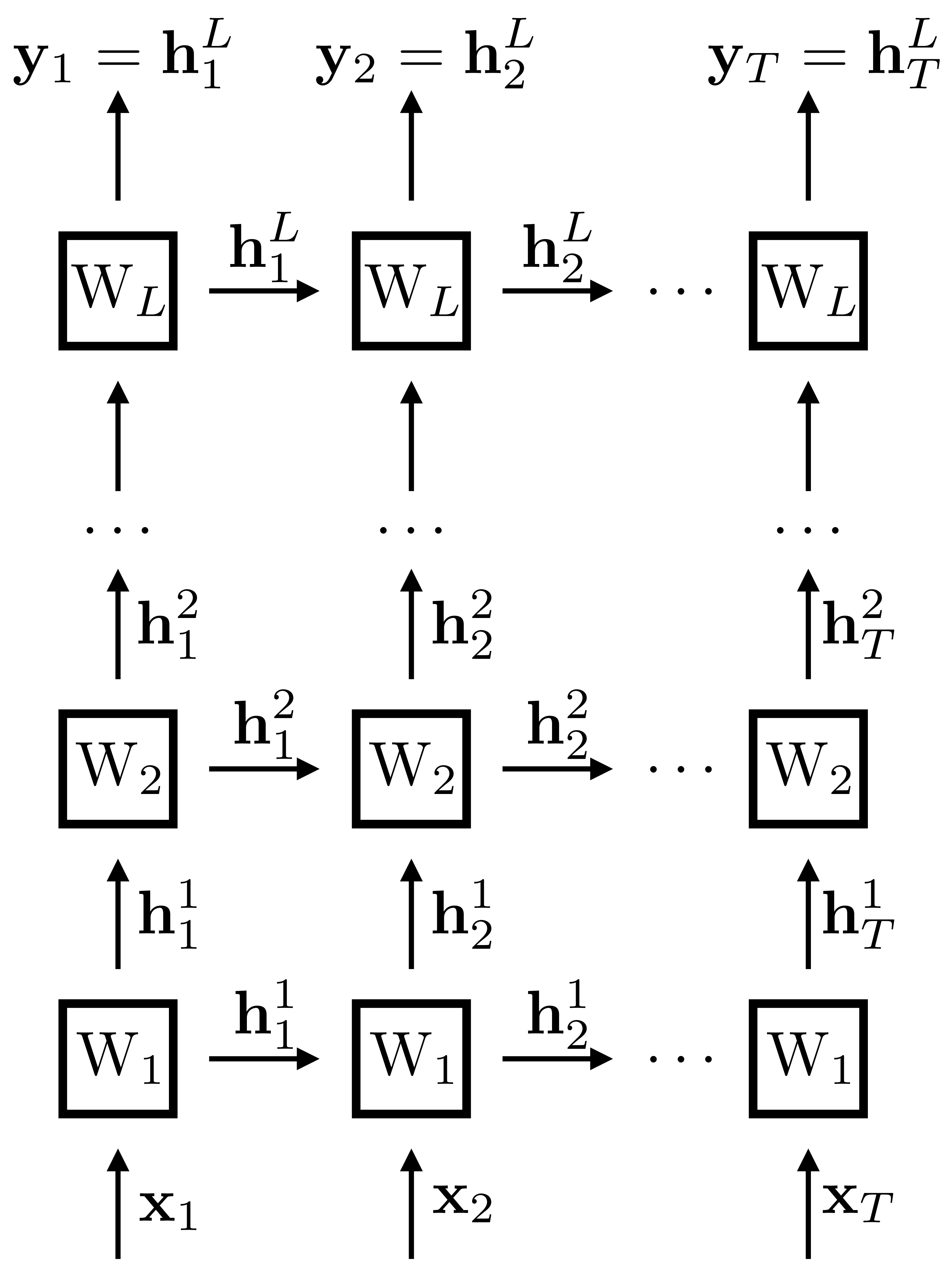
TIME

an ML Frontier

http://www.asimovinstitute.org/neural-network-zoo/

Observe State

Recurrent Neural Networks



Shun-Ichi Amari. IEEE Trans. on Systems Man and Cybernetics.(1972)
J.HopfieJd. Neural Networks as Physical Systems with Emergent Collective Computational Abilities. PNAS, 1982

$$\mathrm{W}_k = \{W_{k,F}, W_{k,R}\}$$
$$\mathbf{h}_t^k = \mathrm{W}_{k,F}\mathbf{h}_t^{k-1} + \mathrm{W}_{k,R}\mathbf{h}_{t-1}^k$$

- Updating the recurrent weights requires computing how the **error on a given step depends on all previous activations**.

KEY PROBLEM : The Vanishing Gradient

Pineda, PRL, 1989

Takens Theorem (1981):

Phase space reconstruction & delay coordinates embeddings

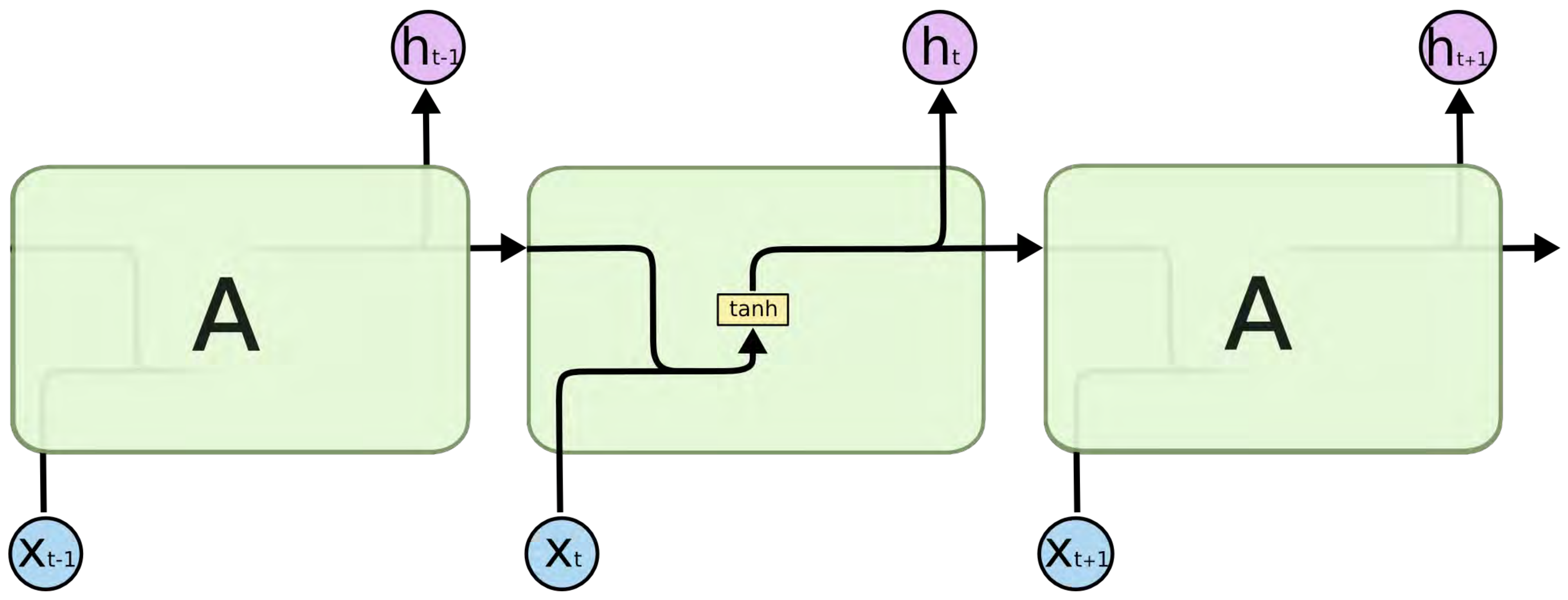
$$\frac{d\xi_t}{dt} = \tilde{F}(\xi_t, \xi_{t-\tau}, \xi_{t-2\tau}, \ldots, \xi_{t-k\tau})$$

Lose effectiveness if observations are scarce

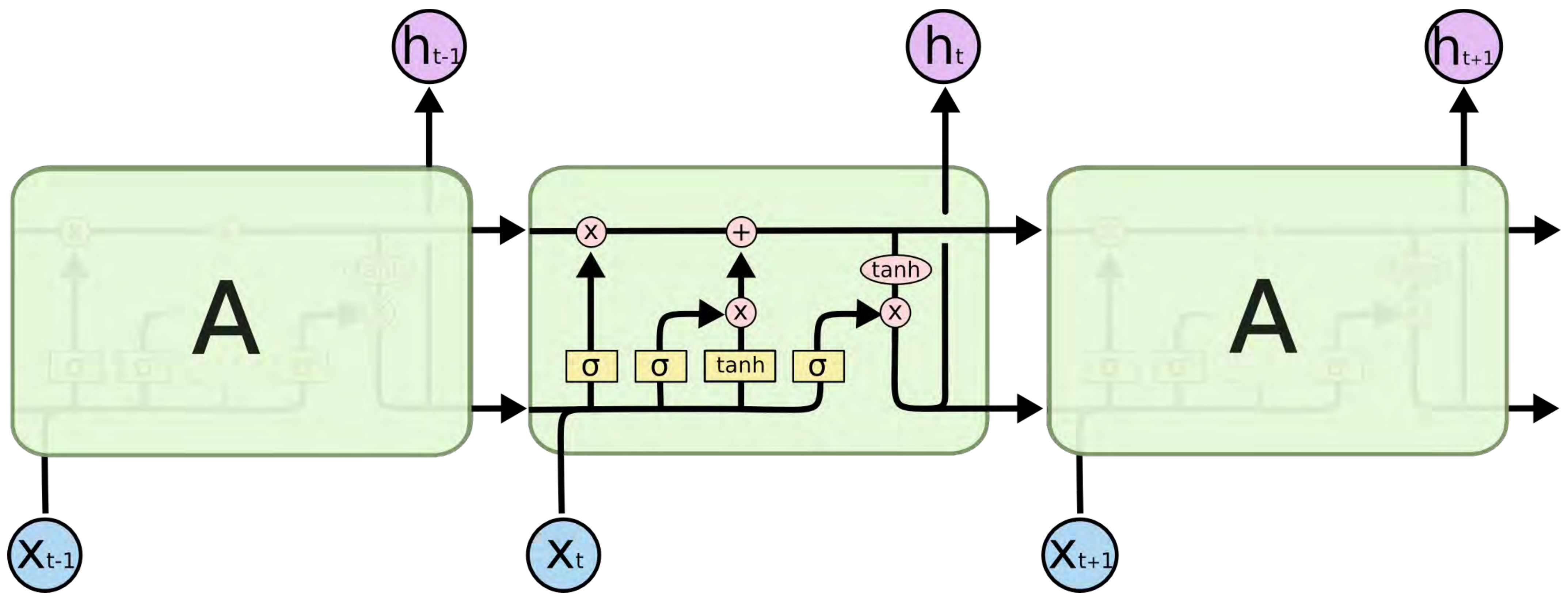
Long Short Term Memory Networks

Schmidhueber and Hochreiter, 1998

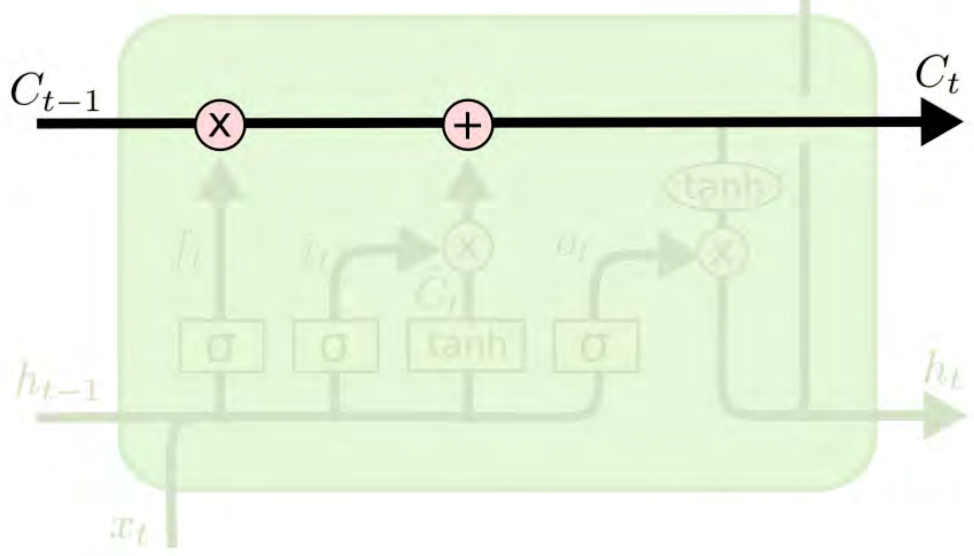
Normal RNN:

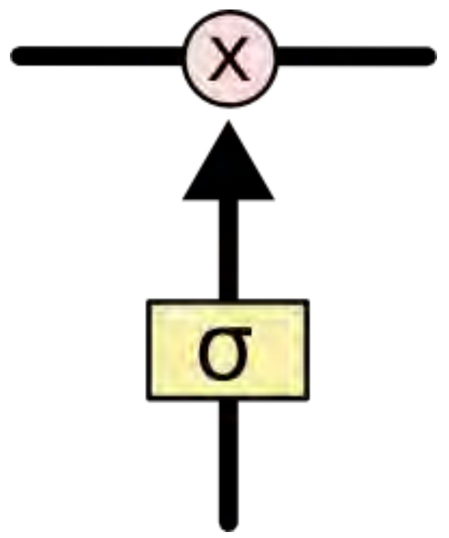


LSTM:



cell's memory





- LSTM cells regulate access to the cell's memory.
- Memory access regulated by dedicated layers with sigmoid non-linearity.
- **KEY IDEA: dependency of output from previous cell's memory does not depend on a chain of matrix multiplications**

Reduced Order Models : Stochastic and Data Driven Corrections

Add noise and propagate uncertainty

$$\frac{d\xi}{dt} = L_{\Phi_1}\xi + H_{\Phi_1}(\xi) + \theta(\omega, t)$$

- Mean Stochastic Model (MSM)
- Gaussian process (GP)
- ….

Uncertainty often grows too fast for radically truncated systems

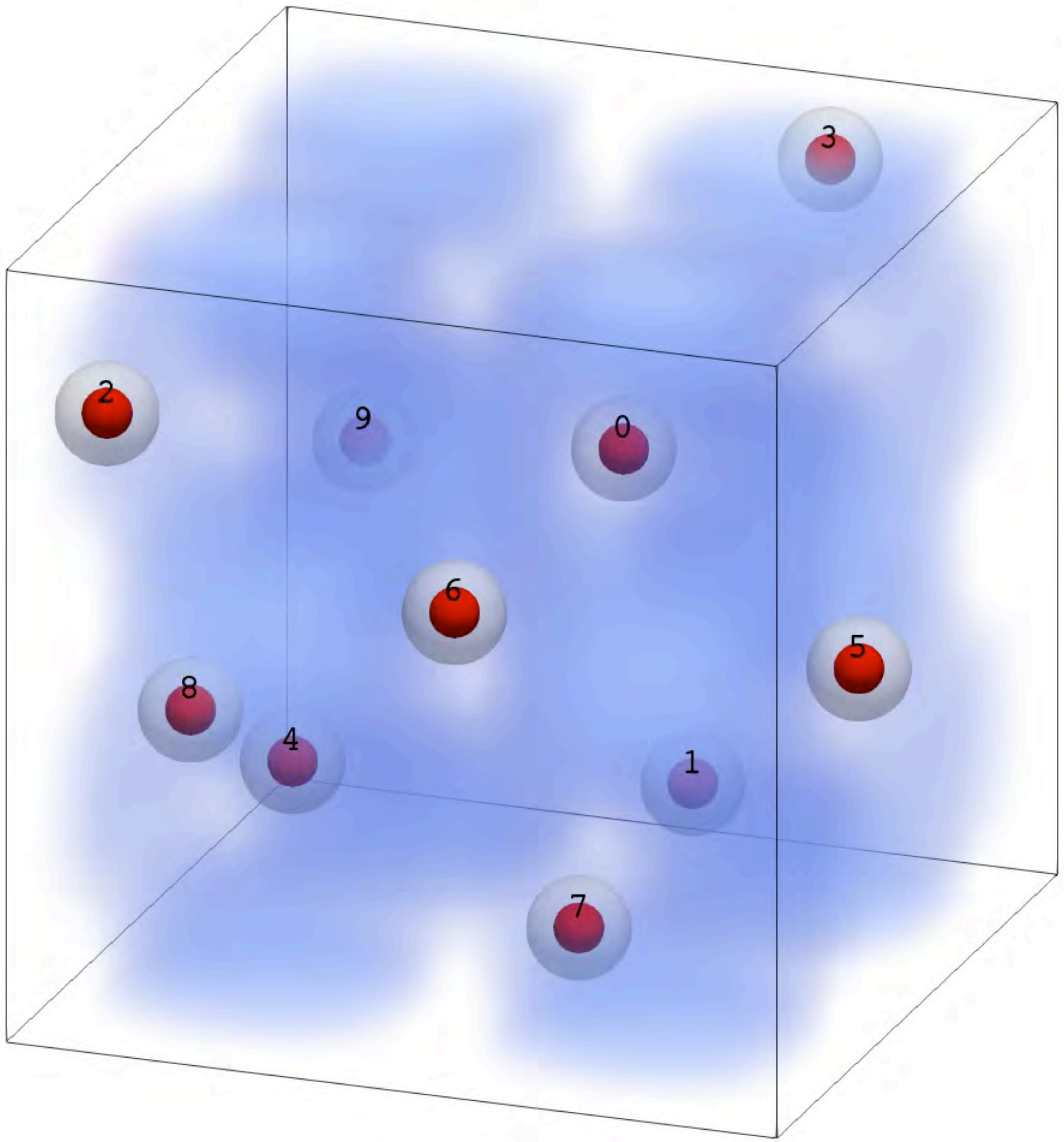
Phase space reconstruction & delay coordinates embeddings (Takens Theorem, 1981)

$$\frac{d\xi_t}{dt} = \tilde{F}(\xi_t, \xi_{t-\tau}, \xi_{t-2\tau}, \dots, \xi_{t-k\tau})$$

Lose effectiveness if observations are scarce

Kinematic Models for Bubbles Trained with DNS

Re = 400



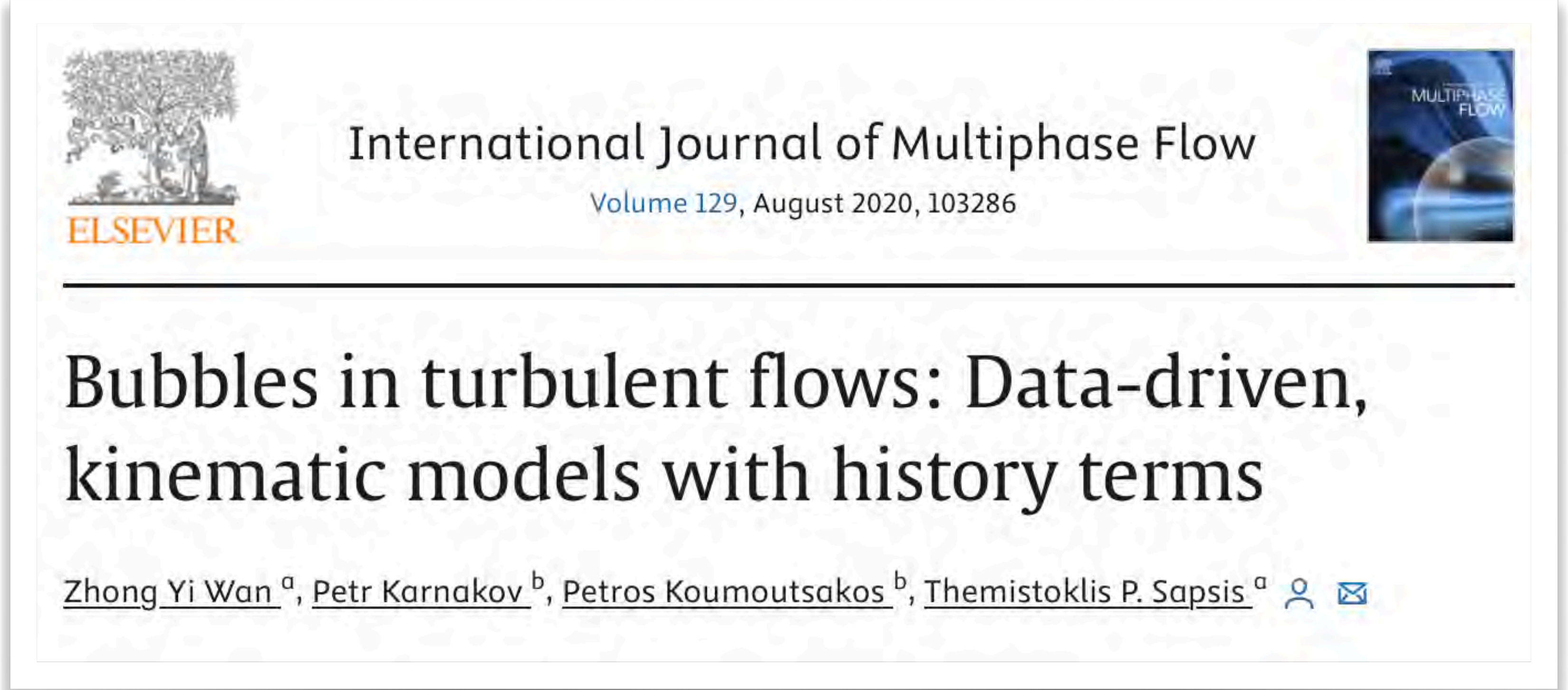
- Two-component incompressible flow with volume fraction α

$$\nabla \cdot \mathbf{u} = 0$$

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}\right) = -\nabla p + \nabla \cdot \mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T) + \mathbf{f}_\sigma$$

$$\frac{\partial \alpha}{\partial t} + (\mathbf{u} \cdot \nabla)\alpha = 0$$

- Finite volume discretization with VoF
- Periodic Taylor-Green vortex initial velocity



ELSEVIER
International Journal of Multiphase Flow
Volume 129, August 2020, 103286

Bubbles in turbulent flows: Data-driven, kinematic models with history terms

Zhong Yi Wan [a], Petr Karnakov [b], Petros Koumoutsakos [b], Themistoklis P. Sapsis [a]

Data-driven kinematic model

Data-driven kinematic model of the form (constant bubble radius, no coalescence)

$$\bar{\mathbf{v}}_t = G(\Theta; \hat{\boldsymbol{\xi}}_t, \hat{\boldsymbol{\xi}}_{t-1}, \hat{\boldsymbol{\xi}}_{t-2}, \ldots)$$

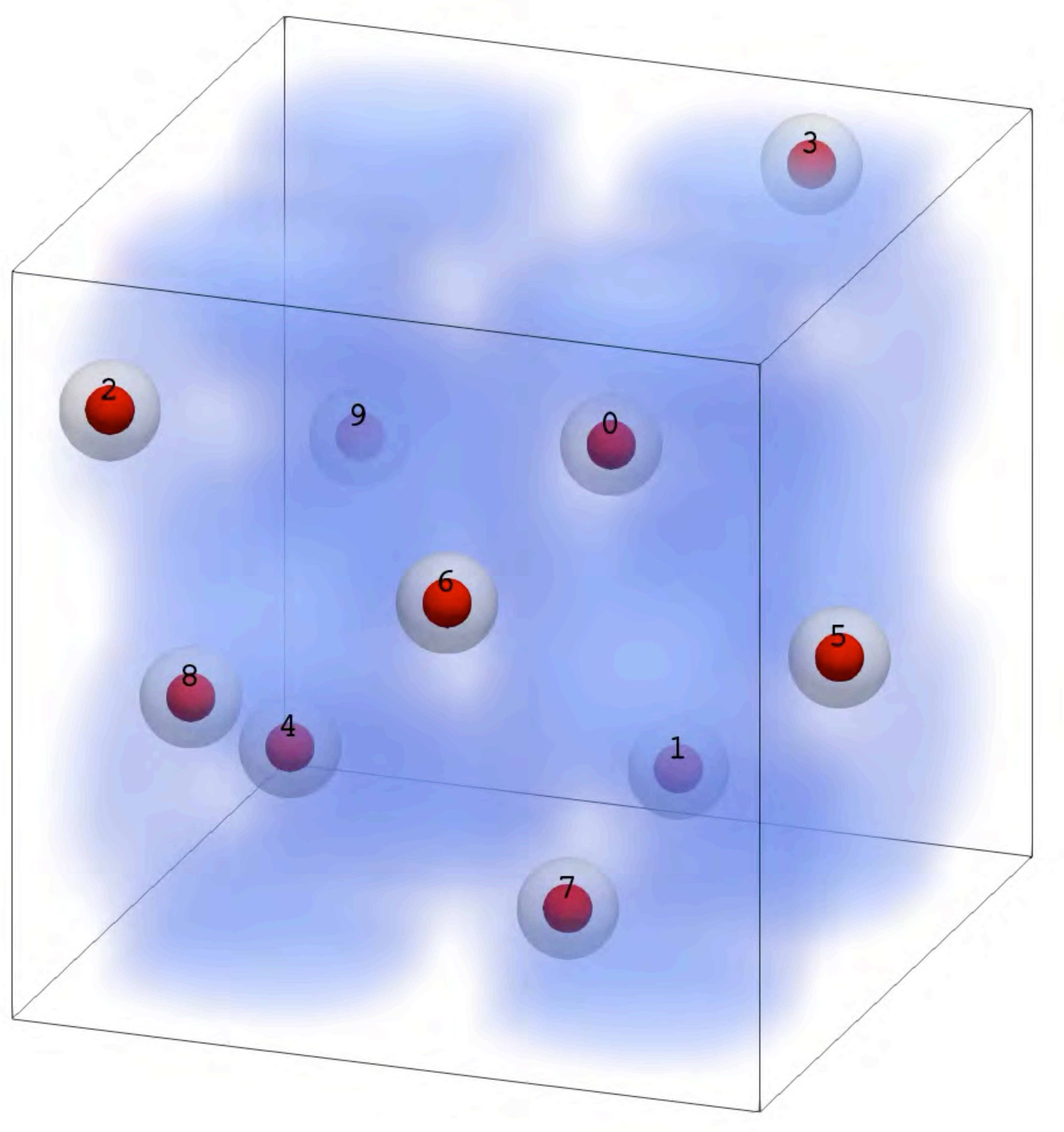
- $\bar{\mathbf{v}}_t$: gas velocity averaged over the entire bubble volume at time t
- Θ : RNN - LSTM parameters to be learned
- $(\hat{\boldsymbol{\xi}}_{\mathbf{t-1}}, \hat{\boldsymbol{\xi}}_{\mathbf{t-1}}, \ldots)$: local flow state experienced by the bubble at time t (e.g. flow velocity, pressure gradient)

$$L(\Theta; \mathscr{D}) = \frac{1}{N}\frac{1}{T}\sum_{n=1}^{N}\sum_{t=1}^{T}\frac{\left\| G\left(\Theta; \hat{\boldsymbol{\xi}}_t^{(n)}, \hat{\boldsymbol{\xi}}_{t-1}^{(n)}, \ldots\right) - \bar{\mathbf{v}}_t^{(n)} \right\|}{\mathrm{var}(v)}$$

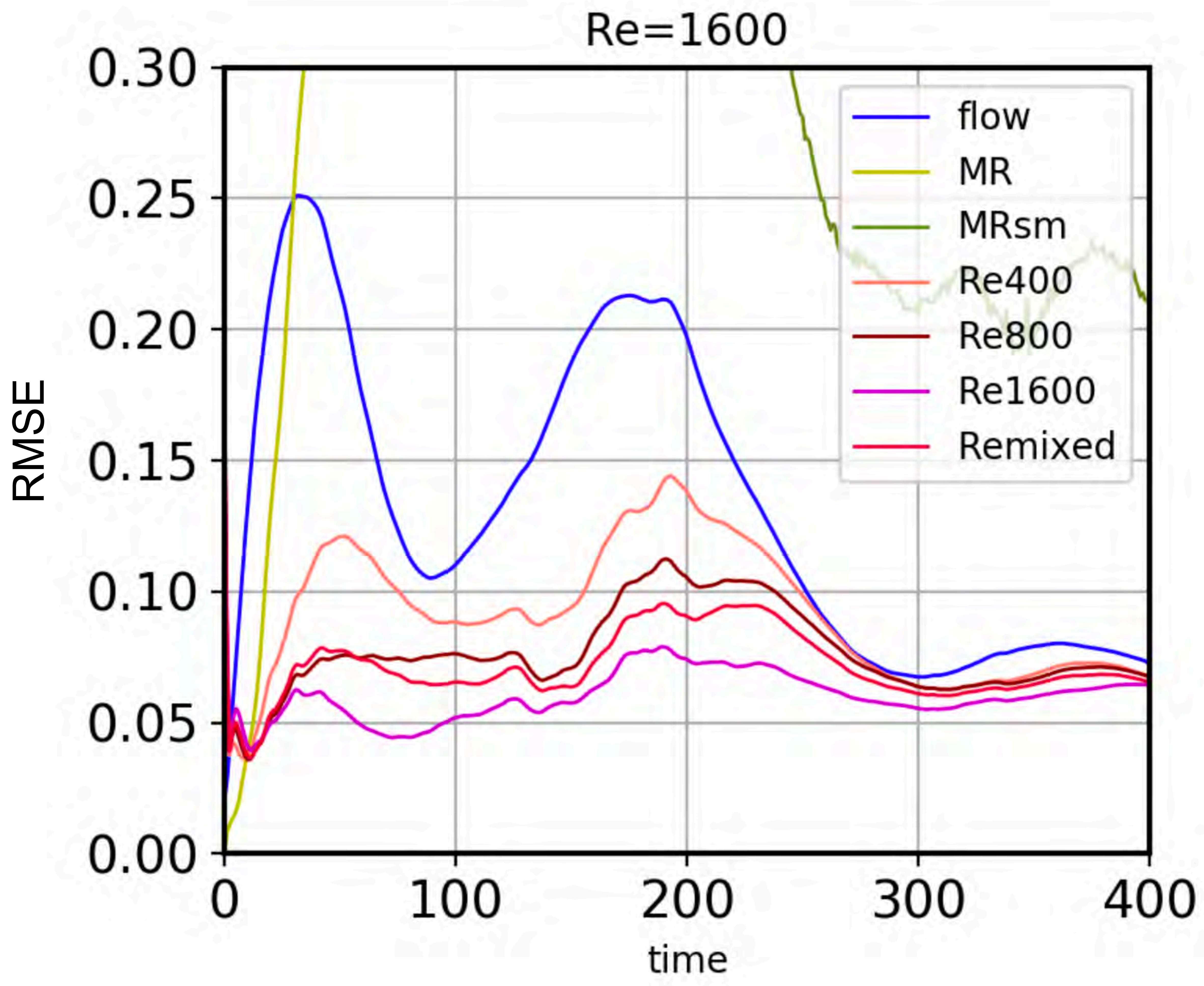
computed over DNS-simulated dataset $\mathscr{D}$

Data augmentation using symmetries

Prediction error for bubble velocity



Re = 1600



- Maxey-Riley (**MR**) $\dot{\mathbf{v}} = \frac{1}{\epsilon}(\mathbf{u} - \mathbf{v}) + \frac{3\gamma}{2}\frac{D\mathbf{u}}{Dt}$
- slow-manifold approximation (**MRsm**) $\mathbf{v} = \mathbf{u} + \epsilon\left[\frac{3\gamma}{2} - 1\right]\frac{D\mathbf{u}}{Dt}$
- **RNNs using training data sets:**
  - **Re400**, **Re800**, **Re1600** - **all mixed**
- average flow velocity $\mathbf{u}$ near bubbles (**flow**)

Results for different Re

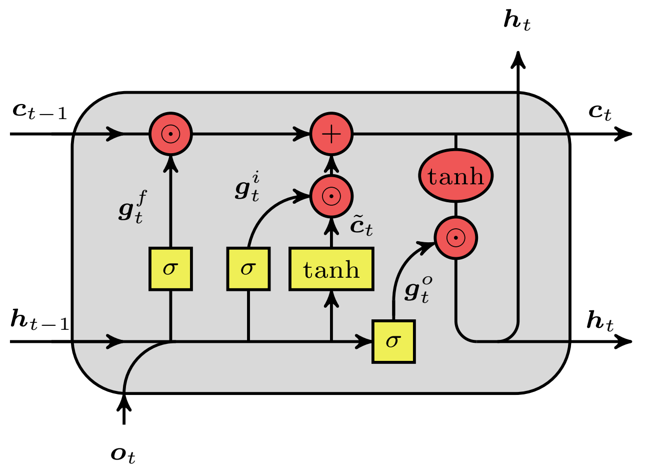
- Better prediction accuracy compared to Maxey-Riley equation for bubble motion
- Model learned at one Re generalizes reasonably well to other Re

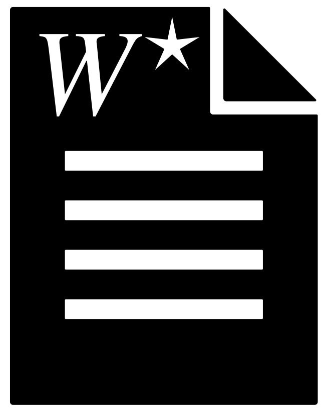
Forecasting on UNSEEN TEMPORAL data

1

TRAIN

&





data *model* *TRAIN* *optimal*

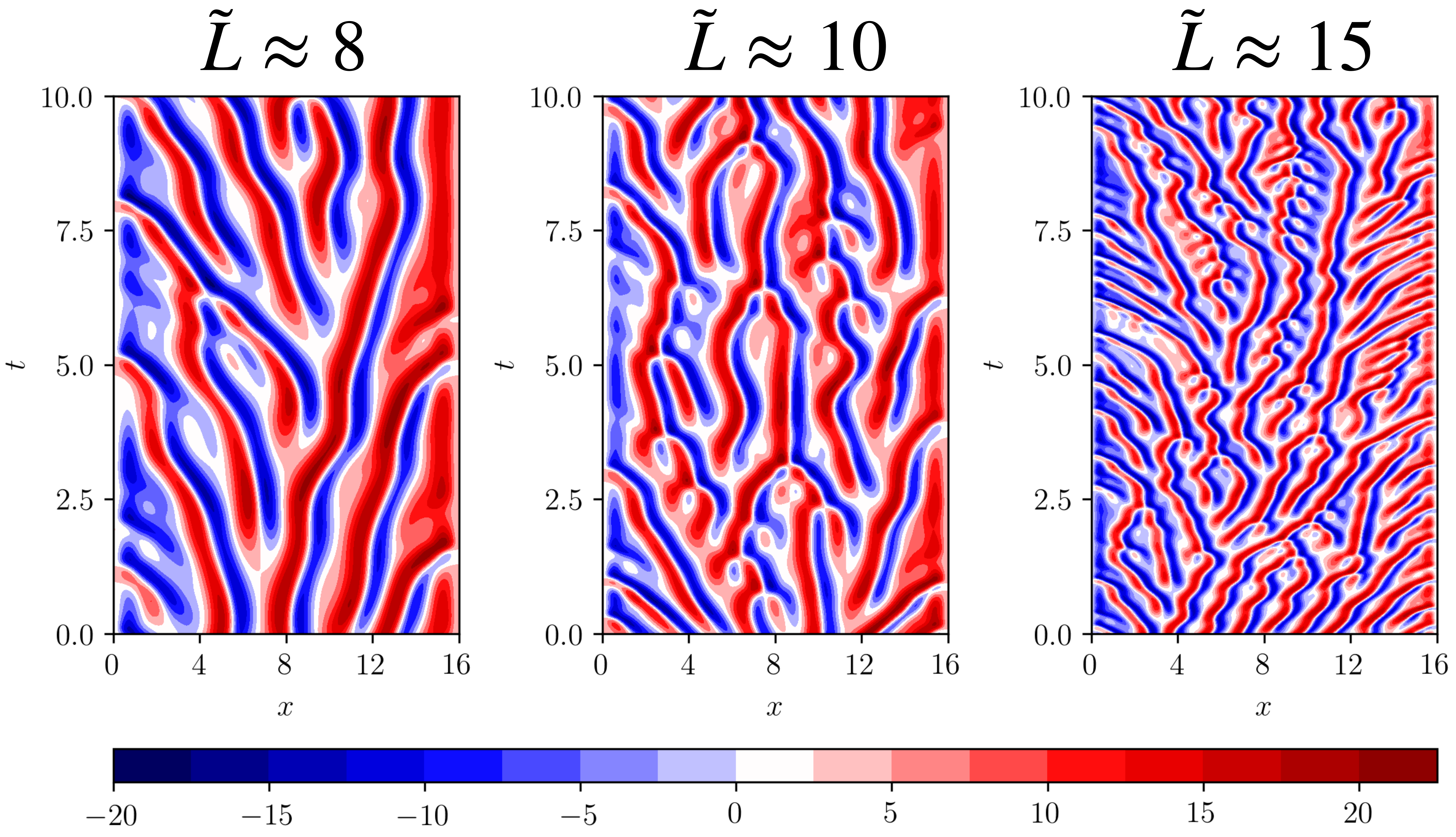
2

TEST

How to predict the dynamics of TEST (unseen) data?

NNs for Dynamical Systems

Kuramoto - Sivashinsky



$$\frac{\partial u}{\partial t} = -\nu \frac{\partial^4 u}{\partial x^4} - \frac{\partial^2 u}{\partial x^2} - u\frac{\partial u}{\partial x},$$

$$u(0,t) = u(L,t) = \left.\frac{\partial u}{\partial x}\right|_{x=0} = \left.\frac{\partial u}{\partial x}\right|_{x=L} = 0,$$

$$u(x,0) = u_0(x),$$

$$x \in [0, L] \quad t \in [0, \infty]$$

$$\frac{du_i}{dt} = -\nu \frac{u_{i-2} - 4u_{i-1} + 6u_i - 4u_{i+1} + u_{i+2}}{\Delta x^4} - \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} - \frac{u_{i+1}^2 - u_{i-1}^2}{4\Delta x}$$

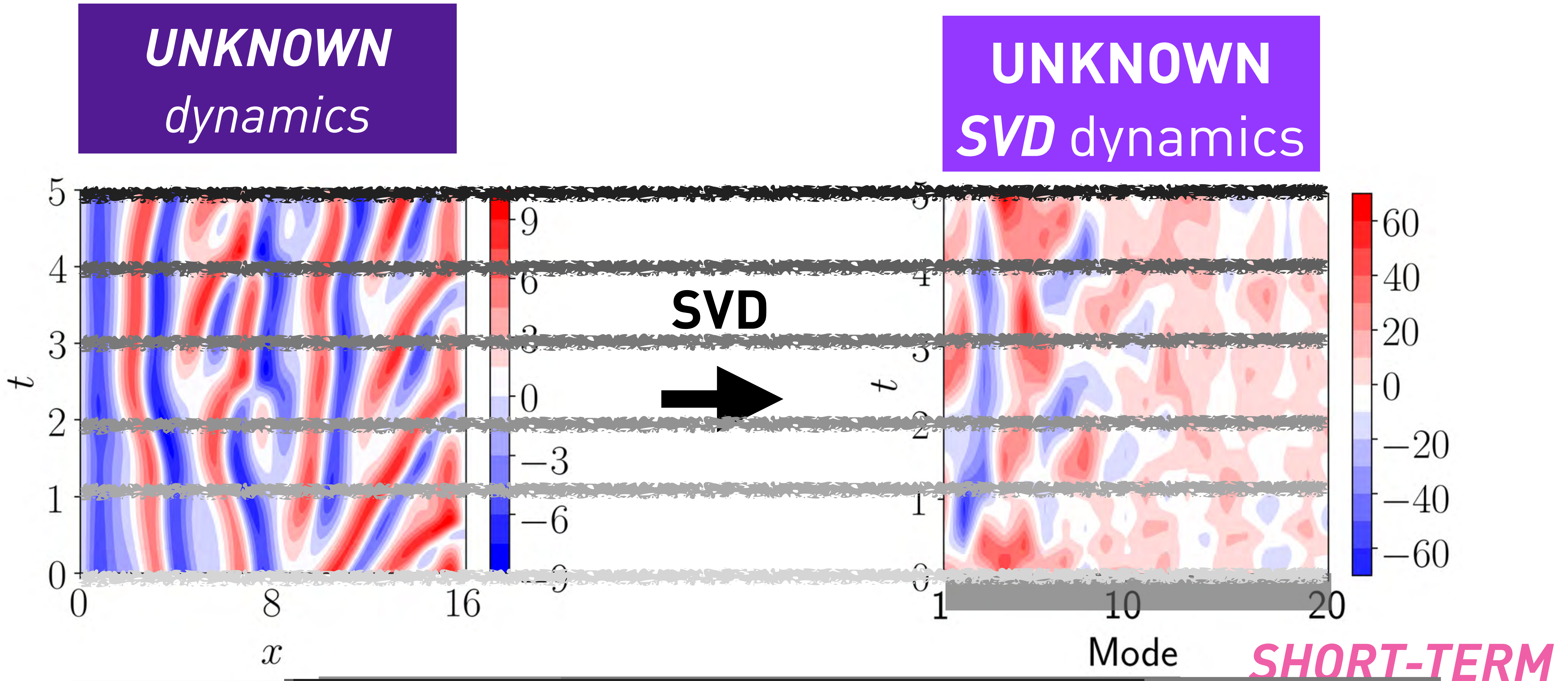
Integration with $dt = 0.02$ up to $T = 10^4$

500.000 **samples**

Forecasting on UNSEEN data

RNNs Train Better with less Data

Train on SVD of the Space

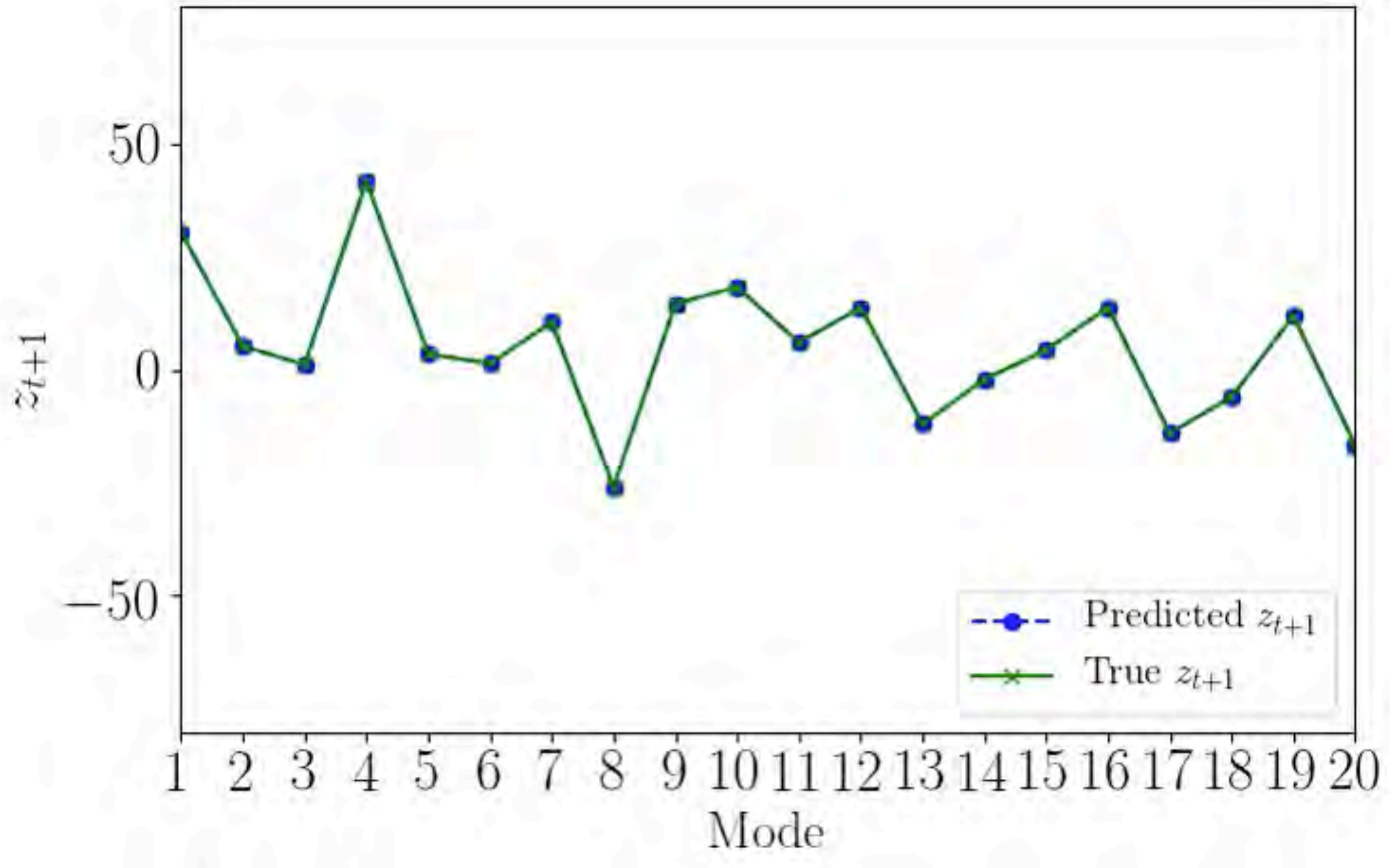


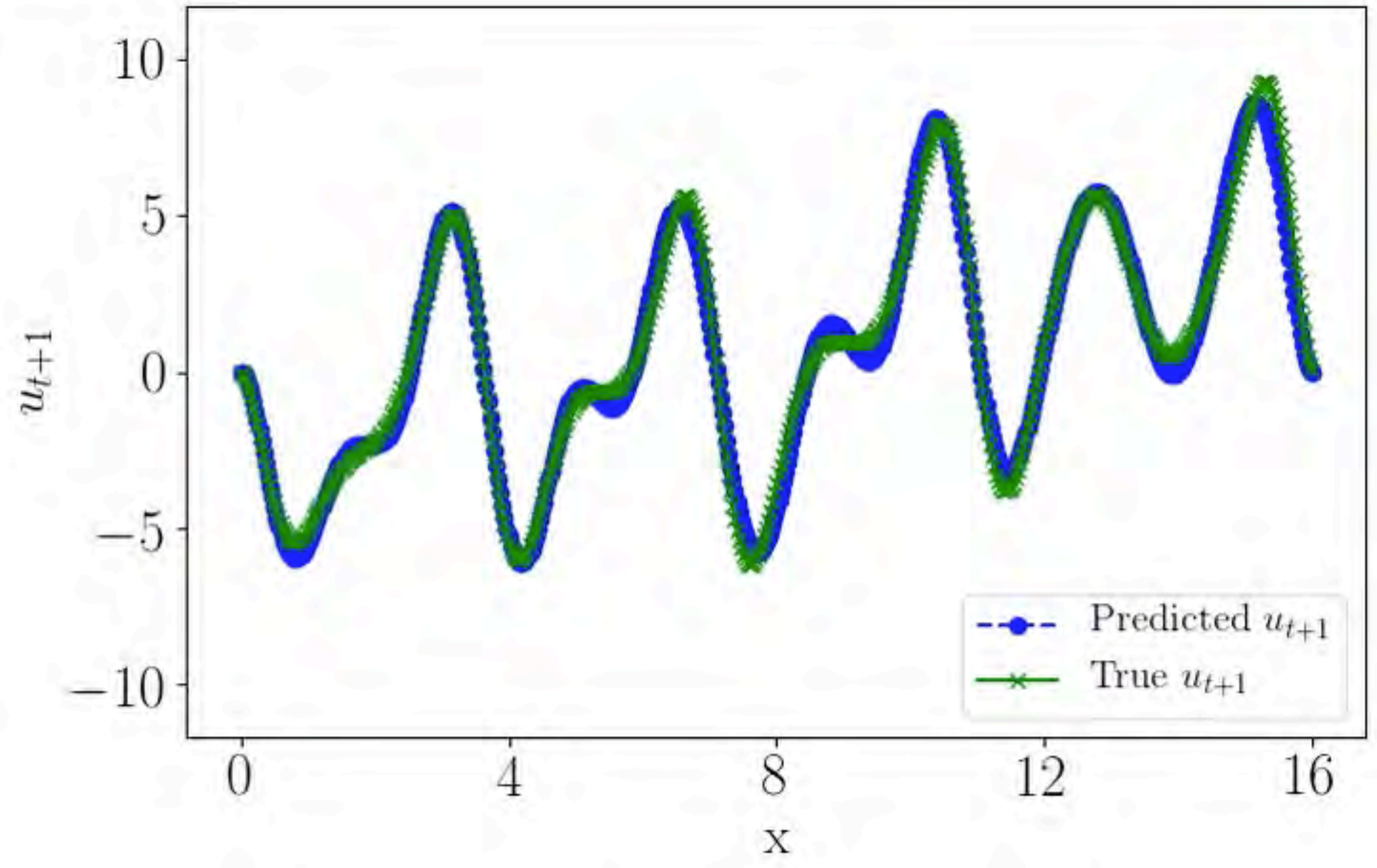
RNNs Fail to Forecast (Later or Earlier)

reduced space

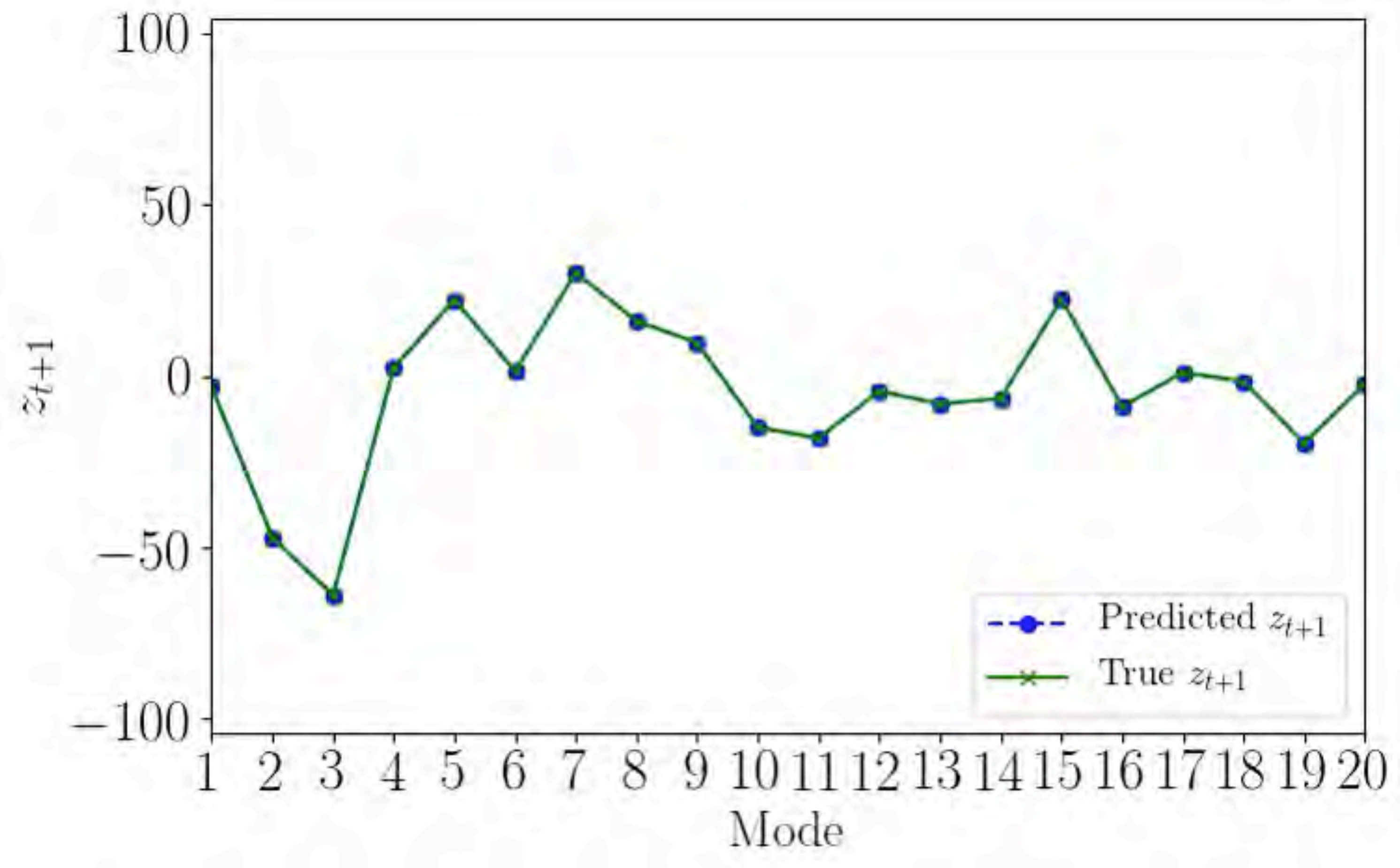
high-dimensional space

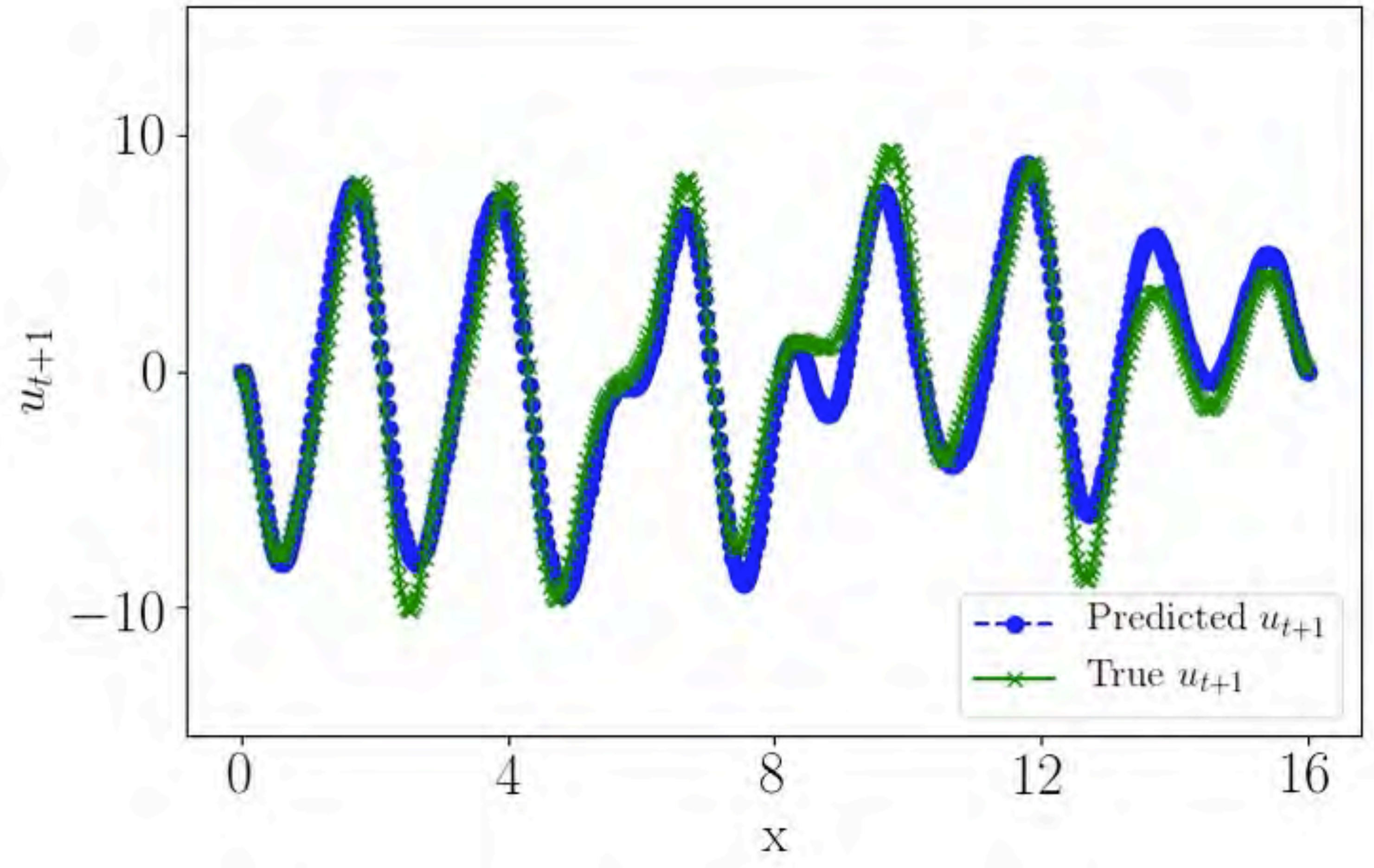
$\tilde{L} \approx 8$





$\tilde{L} \approx 10$





Quanta magazine | Physics | Mathematics | Biology | Computer Science | All Articles

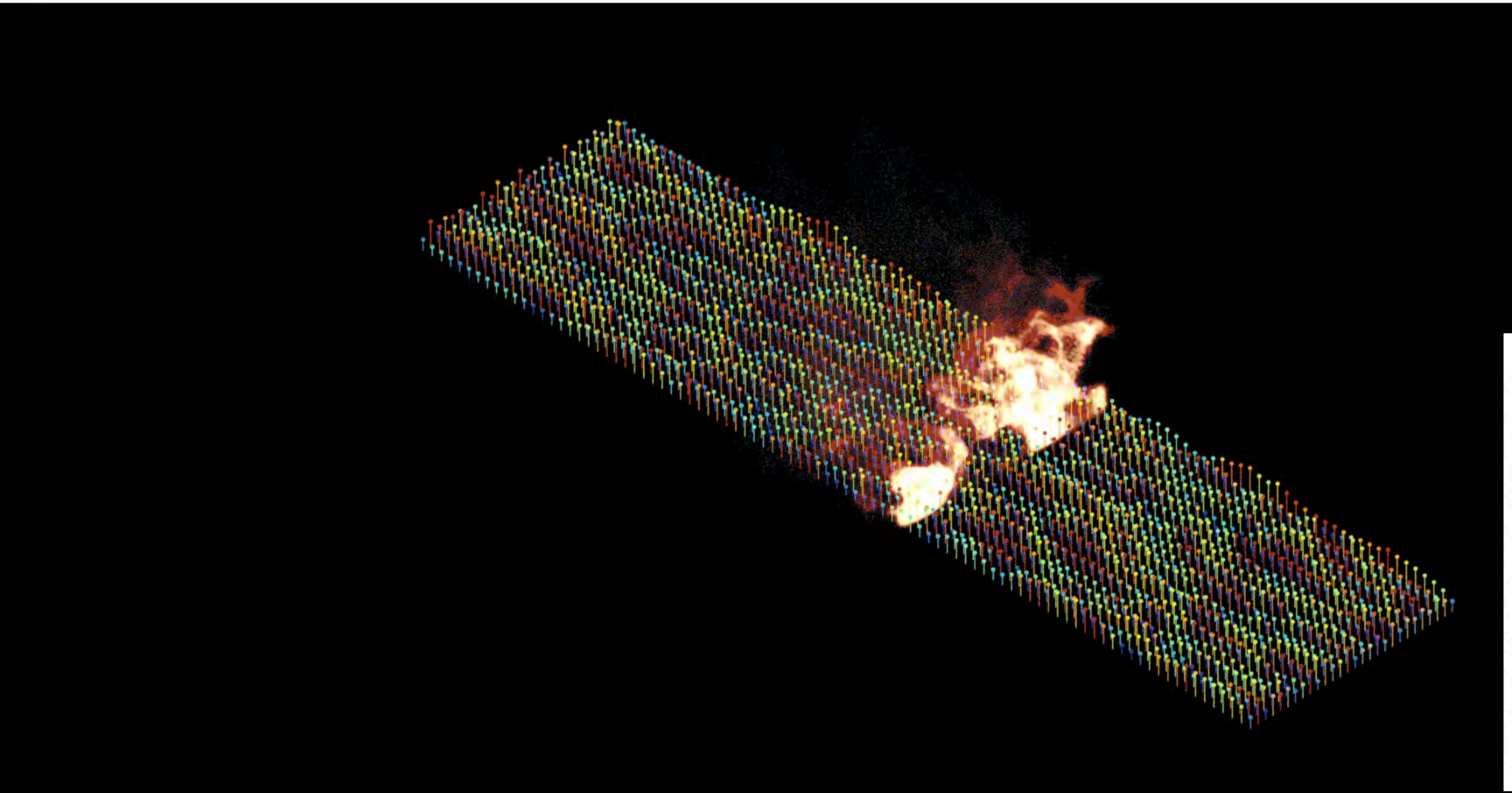
CHAOS THEORY

Machine Learning's 'Amazing' Ability to Predict Chaos

it does not ! (yet ?)

16

In new computer experiments, artificial-intelligence algorithms can tell the future of chaotic systems.



PROCEEDINGS A

rspa.royalsocietypublishing.org

Research

Check for updates

Cite this article: Vlachas PR, Byeon W, Wan ZY, Sapsis TP, Koumoutsakos P. 2018 Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proc. R. Soc. A* **474**: 20170844. http://dx.doi.org/10.1098/rspa.2017.0844

Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks

Pantelis R. Vlachas[1], Wonmin Byeon[1], Zhong Y. Wan[2], Themistoklis P. Sapsis[2] and Petros Koumoutsakos[1]

[1]Chair of Computational Science, ETH Zurich, Clausiusstrasse 33, Zurich, CH-8092, Switzerland

[2]Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA

PLOS | ONE

RESEARCH ARTICLE

Data-assisted reduced-order modeling of extreme events in complex dynamical systems

Zhong Yi Wan[1], Pantelis Vlachas[2], Petros Koumoutsakos[2], Themistoklis Sapsis[1]*

1 Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, United States of America, **2** Chair of Computational Science, ETH Zurich, Zurich, Switzerland

* sapsis@mit.edu

Neural Networks 126 (2020) 191–217

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

ELSEVIER

Check for updates

Backpropagation algorithms and Reservoir Computing in Recurrent Neural Networks for the forecasting of complex spatiotemporal dynamics

P.R. Vlachas[a], J. Pathak[b,c], B.R. Hunt[d,e], T.P. Sapsis[f], M. Girvan[b,c,d], E. Ott[b,c,g], P. Koumoutsakos[a,*]

[a] Computational Science and Engineering Laboratory, ETH Zürich, Clausiusstrasse 33, Zürich CH-8092, Switzerland
[b] Institute for Research in Electronics and Applied Physics, University of Maryland, College Park, MD 20742, USA
[c] Department of Physics, University of Maryland, College Park, MD 20742, USA
[d] Institute for Physical Science and Technology, University of Maryland, College Park, MD 20742, USA
[e] Department of Mathematics, University of Maryland, College Park, MD 20742, USA
[f] Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139-4307, USA
[g] Department of Electrical and Computer Engineering, University of Maryland, MD 20742, USA

CLOSING THOUGHTS

Comment

https://doi.org/10.1038/s42254-024-00726-z

On roads less travelled between AI and computational science

Petros Koumoutsakos

Check for updates

Computational science and artificial intelligence have been drivers and benefactors of advances in algorithms and hardware, each in different ways, and originally with different targets. Petros Koumoutsakos argues that the intellectual space between these two fields is home to exciting opportunities for scientific discovery.

'importance sampling'. There are plenty of opportunities for cross-fertilizing exchanges in algorithms and their applications. Similarly, stochastic and gradient optimization methods have been developed across both communities, but recent works on automatic differentiation indicate that the paths are intersecting again. The emergence and homogenization properties of foundational models that are gaining ground in AI also have counterparts in CoS where emergence is often the outcome of nonlinear differential equations, whereas the concept of homogenization can be recognized for example in particle simulations of phenomena ranging from atoms to galaxies[3]. At the same time the paths of scientific inquiry in AI and CoS may diverge, but I argue that repeated intersection can be exciting. There are many problems where

Nature Reviews Physics, 2024

Learning Effective Dynamics

nature machine intelligence

ARTICLES

https://doi.org/10.1038/s42256-022-00464-w

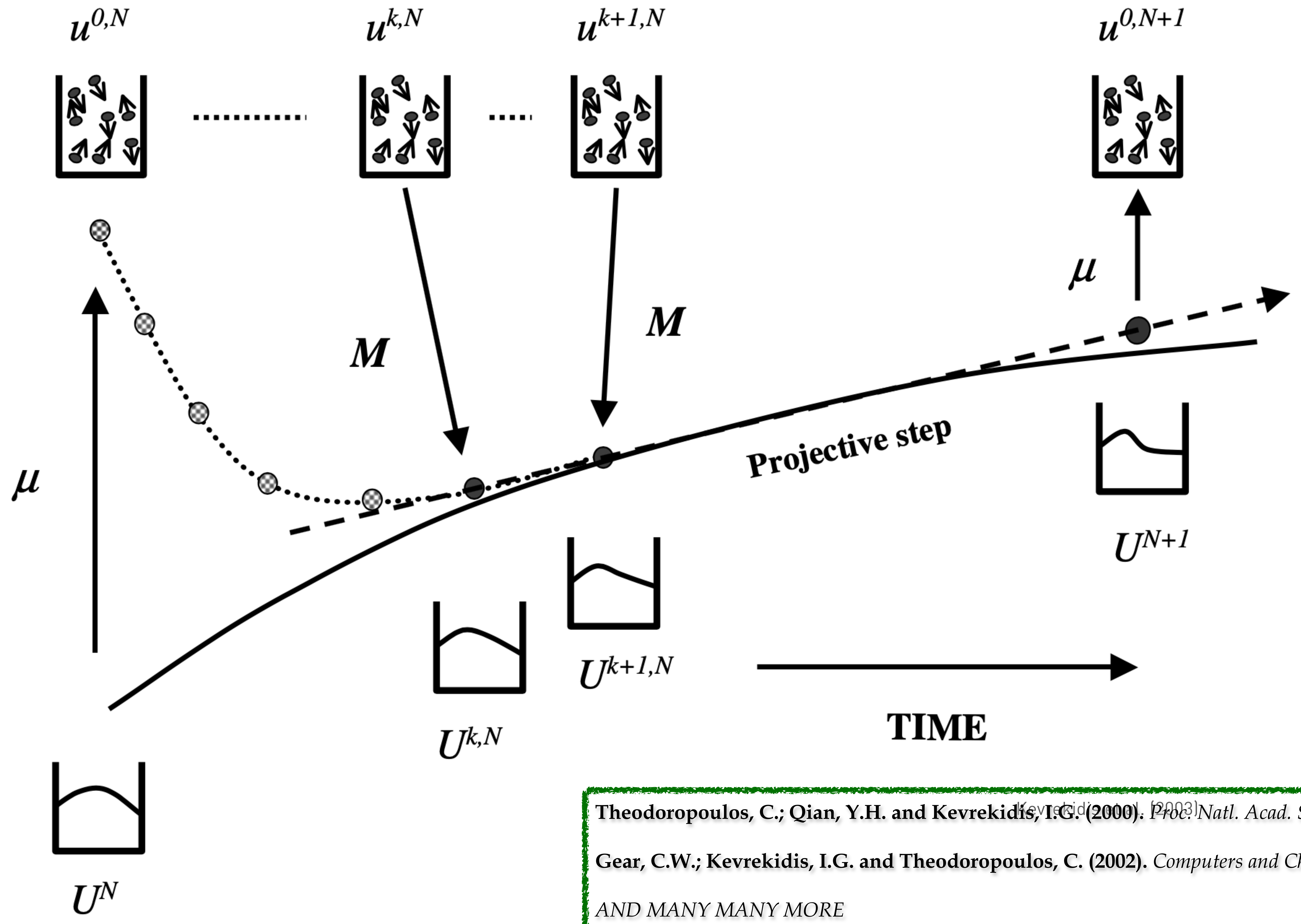
Check for updates

Multiscale simulations of complex systems by learning their effective dynamics

Pantelis R. Vlachas[1,2], Georgios Arampatzis[1,2], Caroline Uhler[3] and Petros Koumoutsakos[1,2]

Equation-Free Framework (EFF) - Kevrekidis,...

Micro-scale model



Macro-scale model

Kevrekidis et al. (2003)

Theodoropoulos, C.; Qian, Y.H. and Kevrekidis, I.G. (2000). *Proc. Natl. Acad. Sci. 97: 9840-9845.*

Gear, C.W.; Kevrekidis, I.G. and Theodoropoulos, C. (2002). *Computers and Chemical Engineering 26: 941-963.*

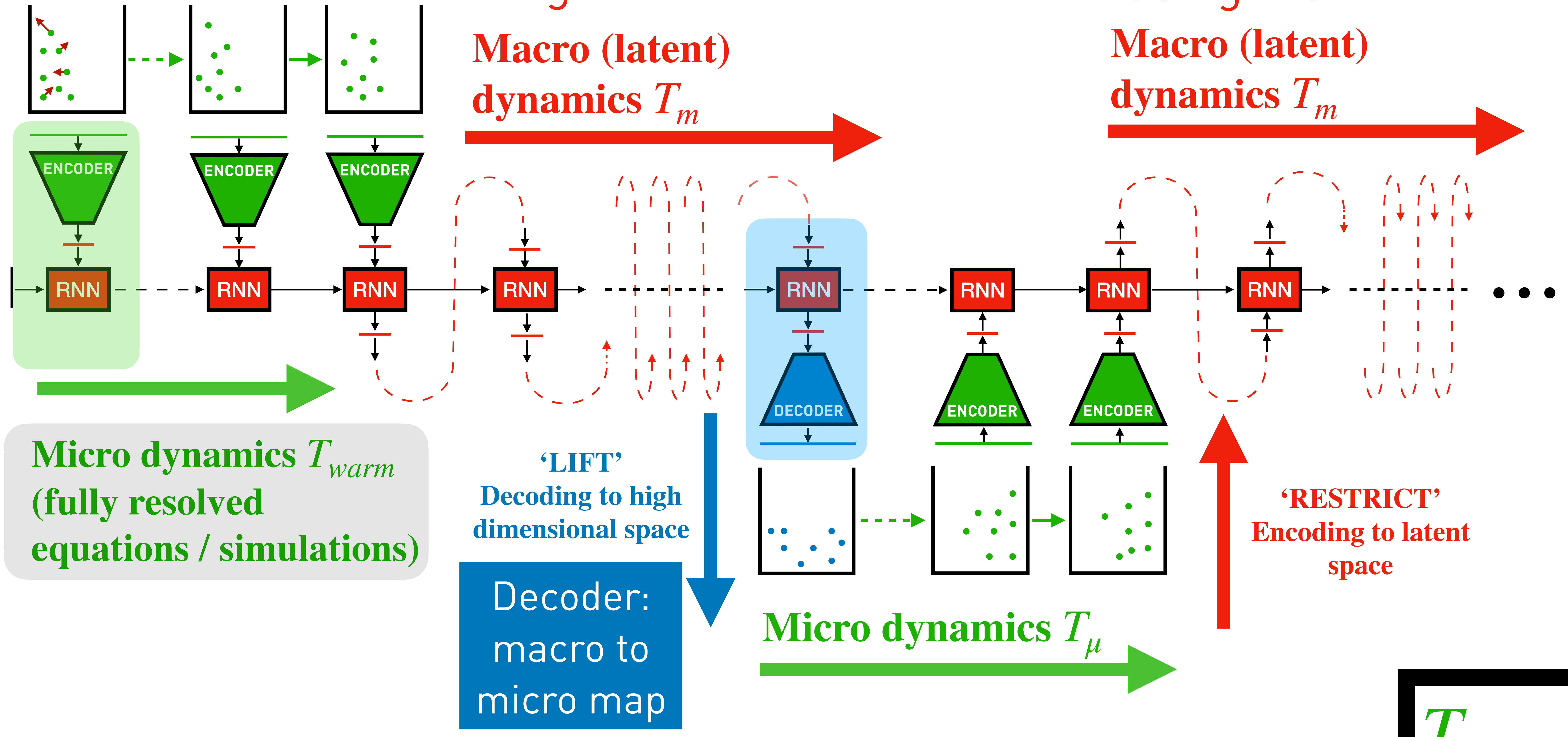
AND MANY MANY MORE

Learning Effective Dynamics

PR Vlachas, G Arampatzis, C Uhler, P Koumoutsakos,
Multiscale Simulations of Complex Systems by Learning their Effective Dynamics,
Nature Machine Intelligence, (2022)

Update the macro dynamics using the RNN

Update the macro dynamics using the RNN



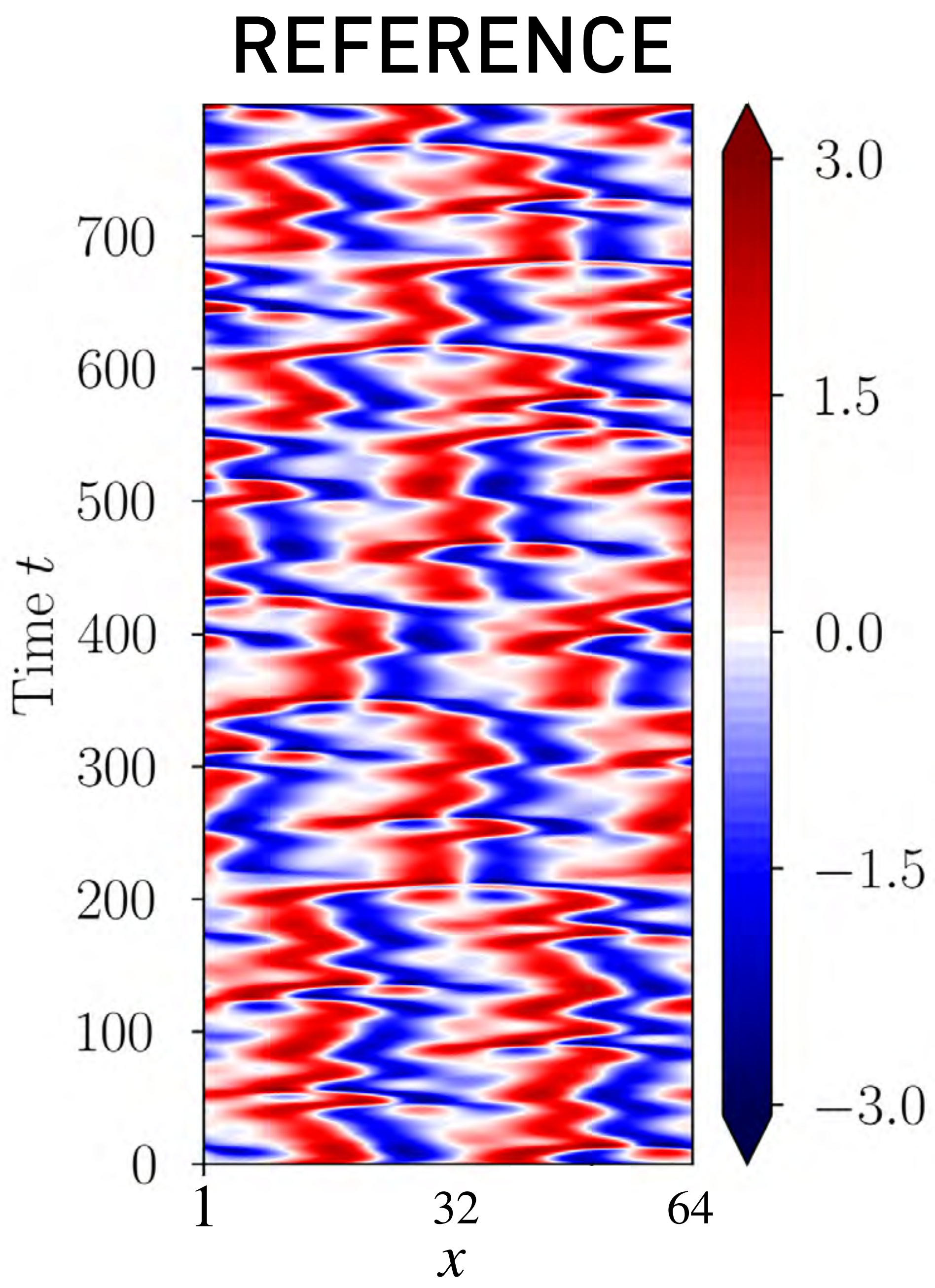
$$\boxed{T_{warm}, T_\mu \ll T_m}$$

Update the micro dynamics and Train RNN

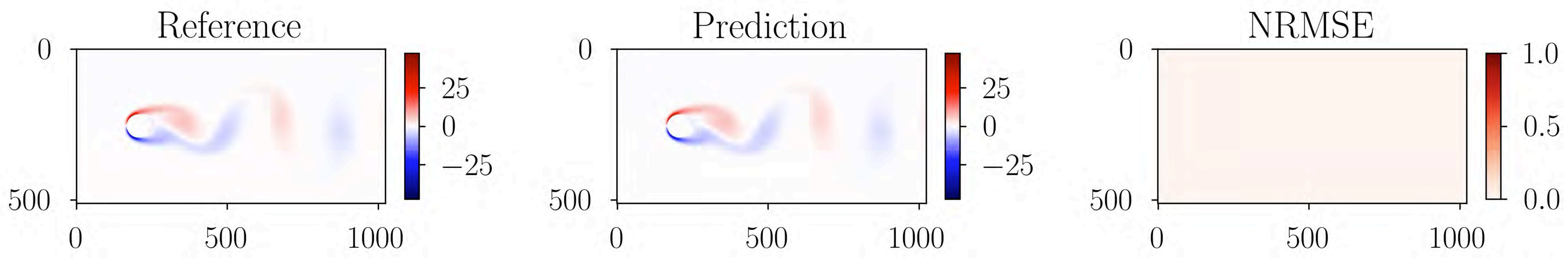
Update the micro dynamics (and continue training of RNN)

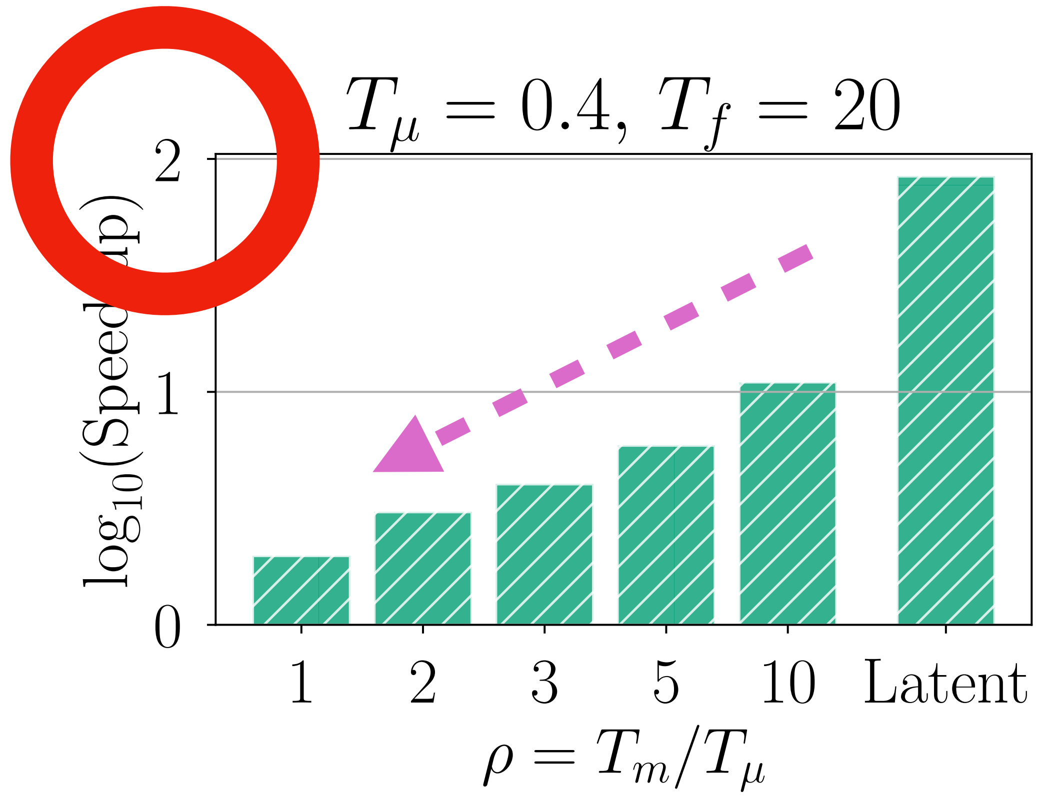
Kuramoto-Sivashinsky $L = 22$

PR Vlachas, G Arampatzis, C Uhler, P Koumoutsakos,
Multiscale Simulations of Complex Systems by Learning their Effective Dynamics,
Nature Machine Intelligence, (2022)



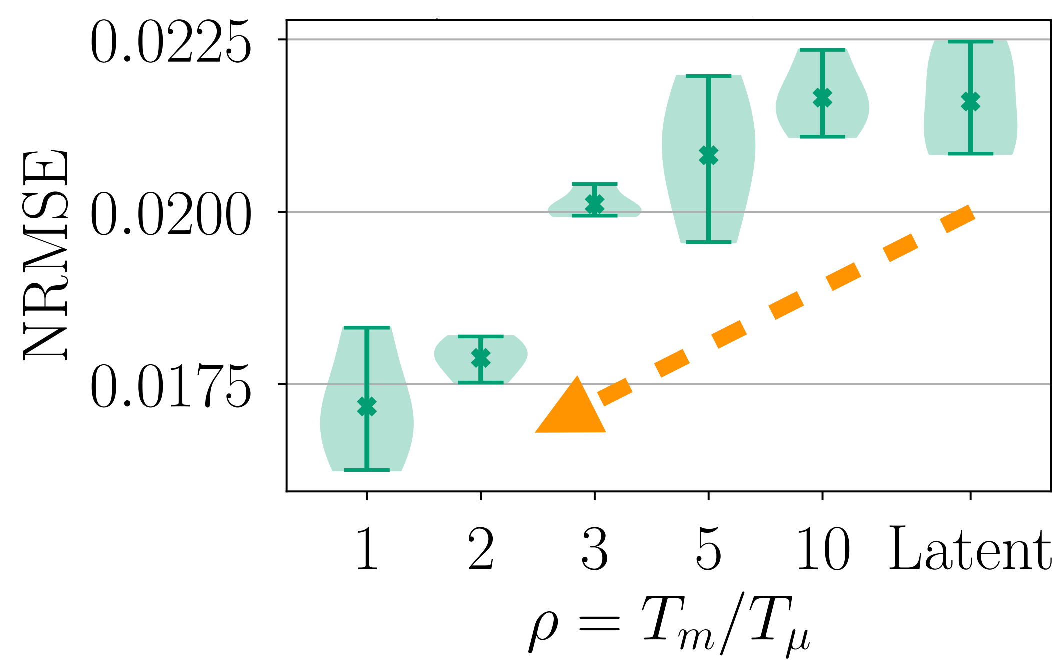
Cylinder at $Re = 100$ - (LED $d_z = 2$)

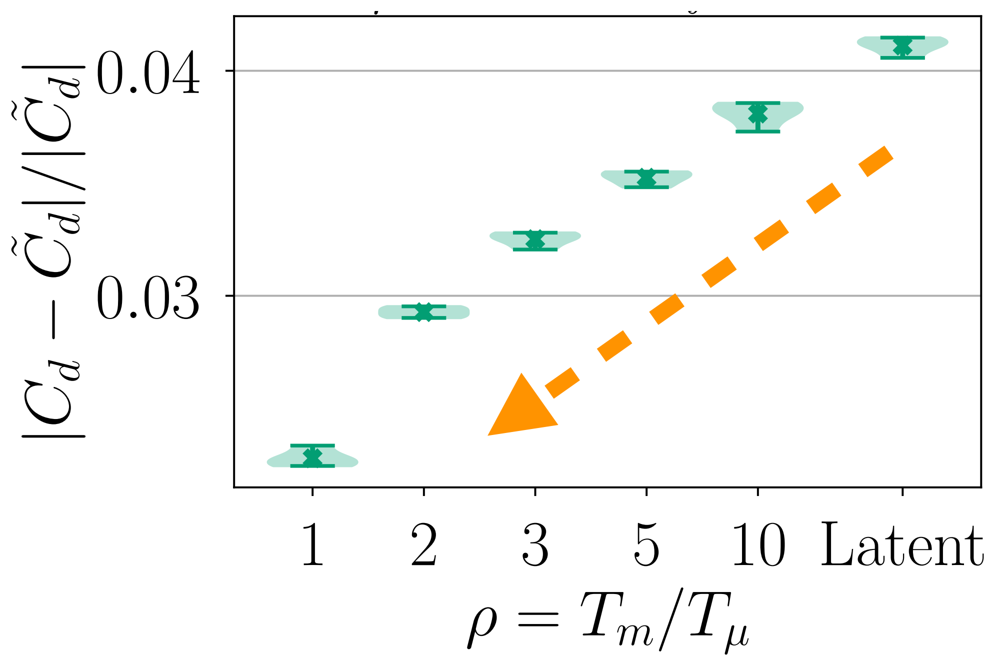




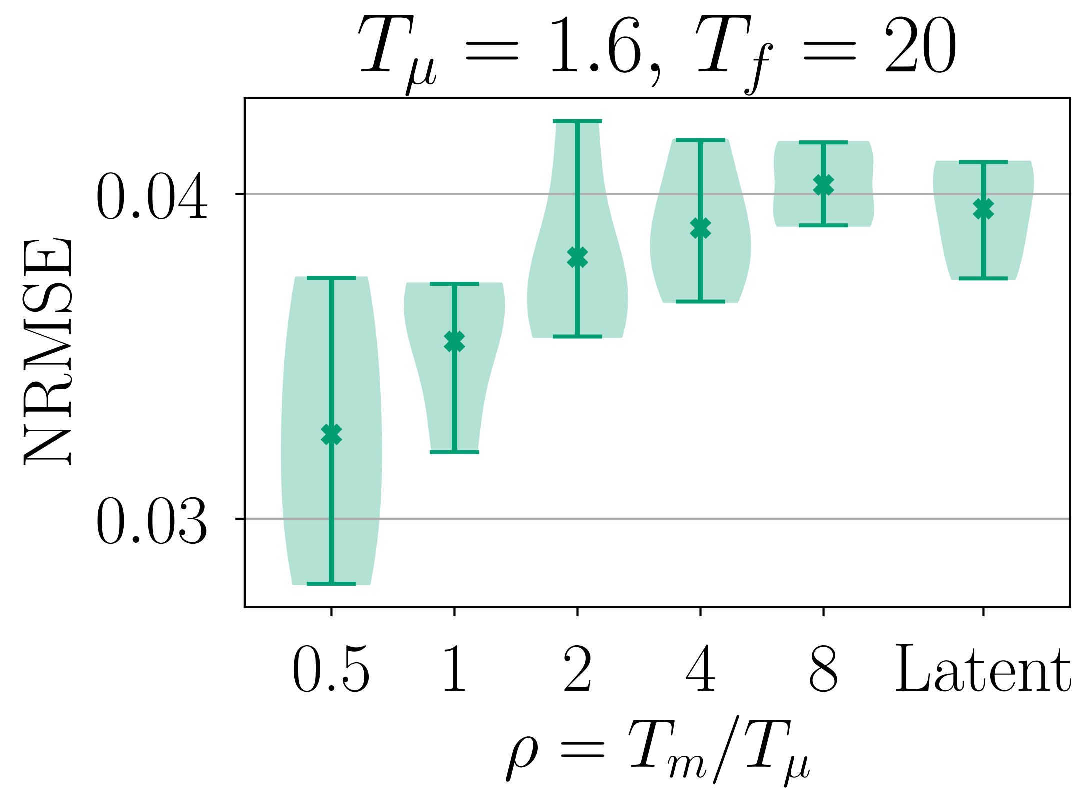
- Micro solver: Finite Differences solver (CubimUP2D) employing 12 cores
- State: velocity in x- and y- direction and pressure $\mathbf{s}_t \in \mathbb{R}^{3\times512\times1024}$
- LED with latent dimension of $d_\mathbf{z} = 2$, $\Delta t = 0.2$

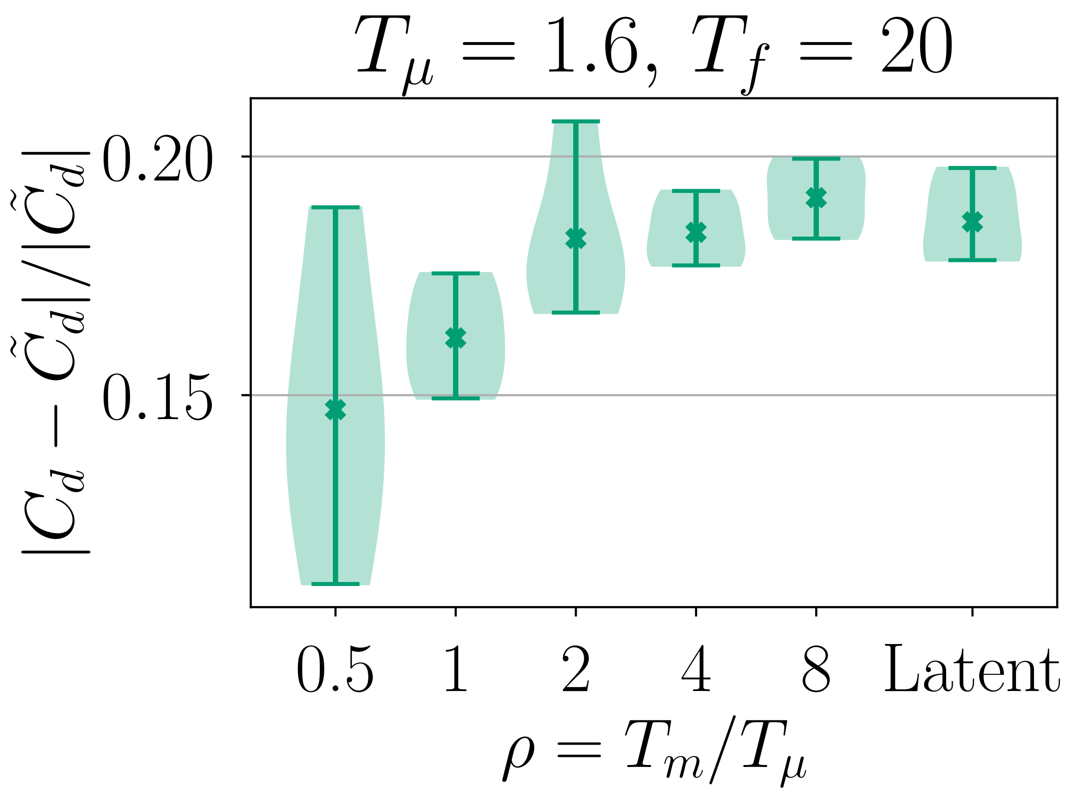
- LED captures **long-term evolution** of velocity and pressure fields (low NRMSE)
- LED is up to **two orders of magnitude** faster than CubismUP2D
- Recovers drag coefficient with $\approx 2-4\,\%$ error

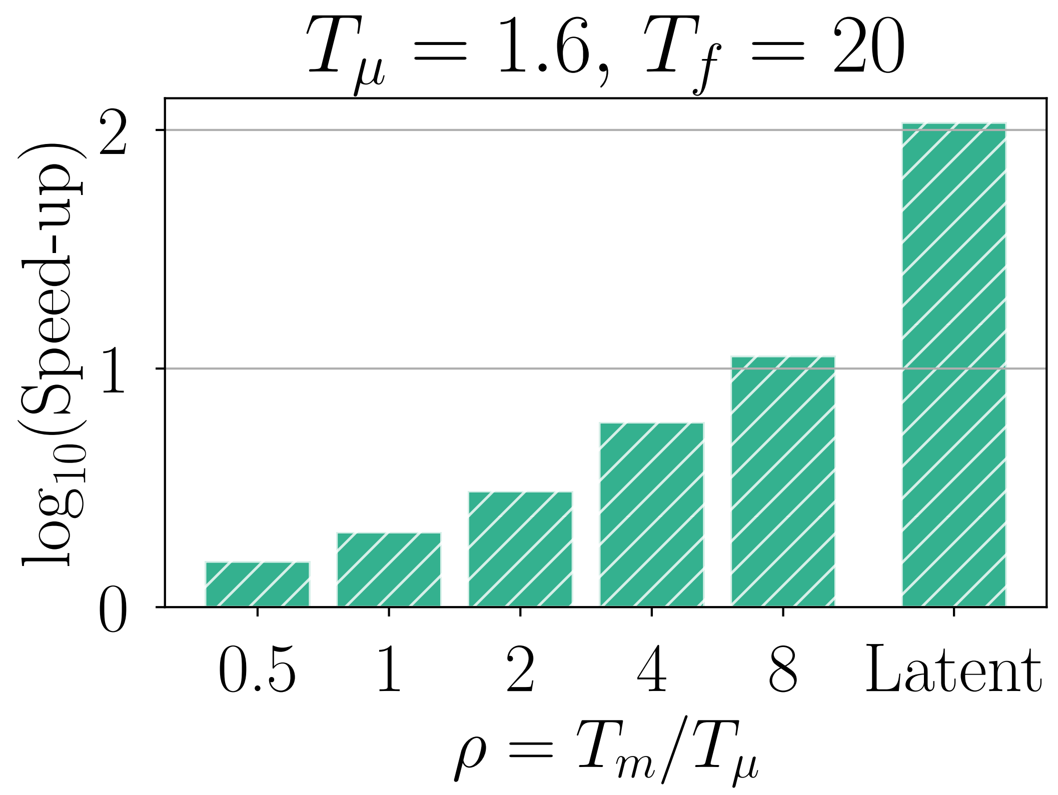




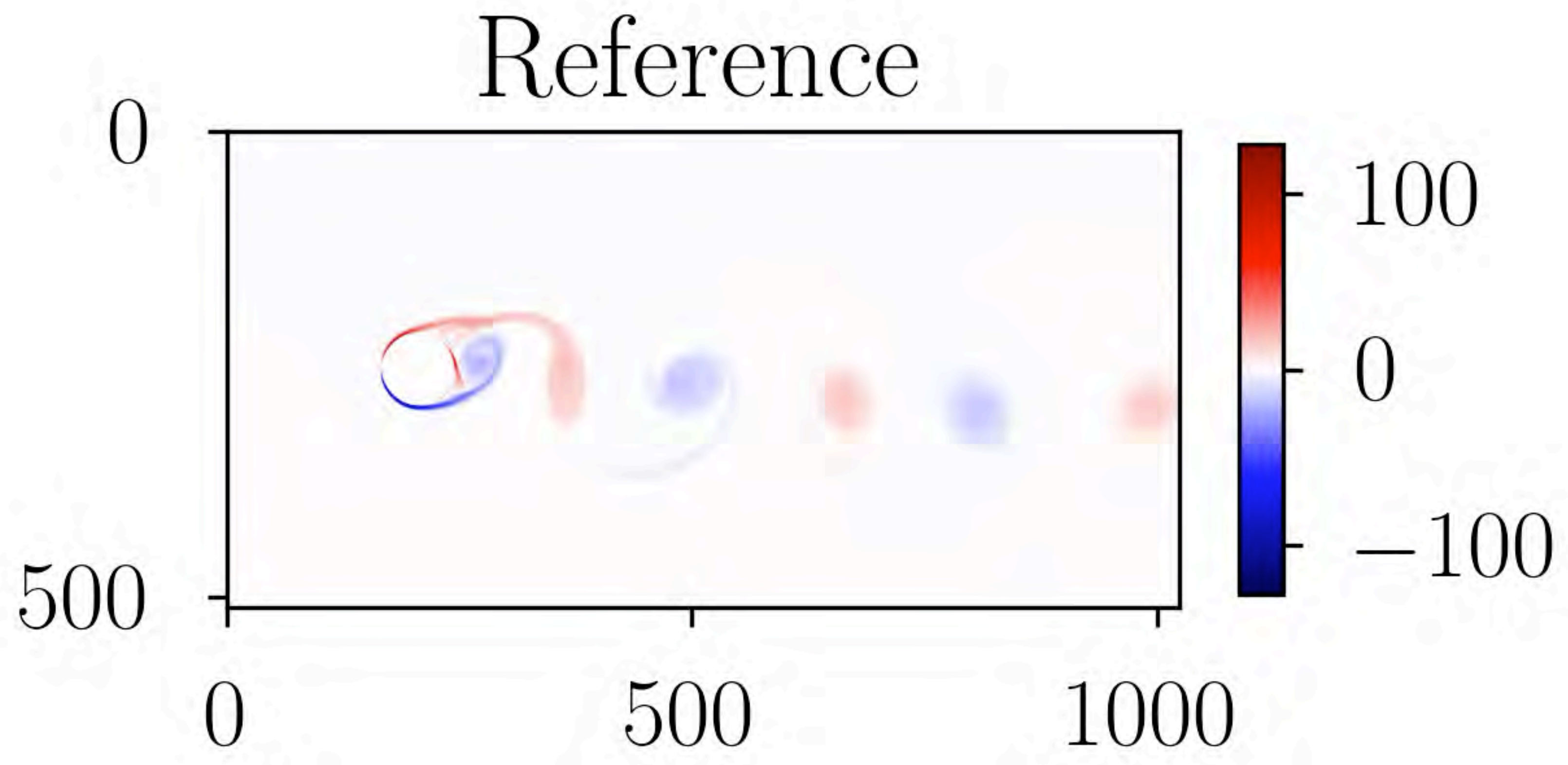
Cylinder at $Re = 1000$ (LED $d_z = 10$)

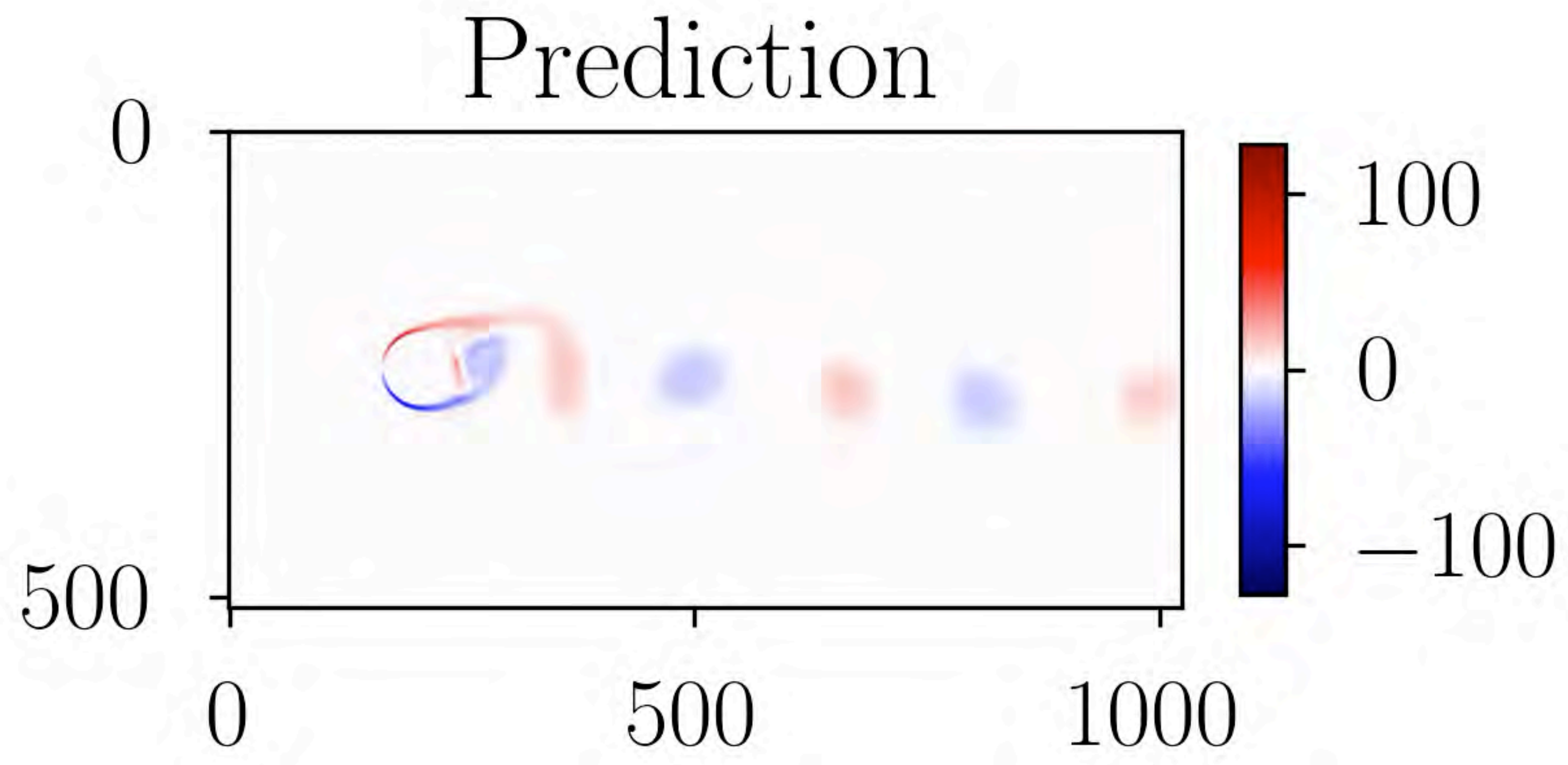


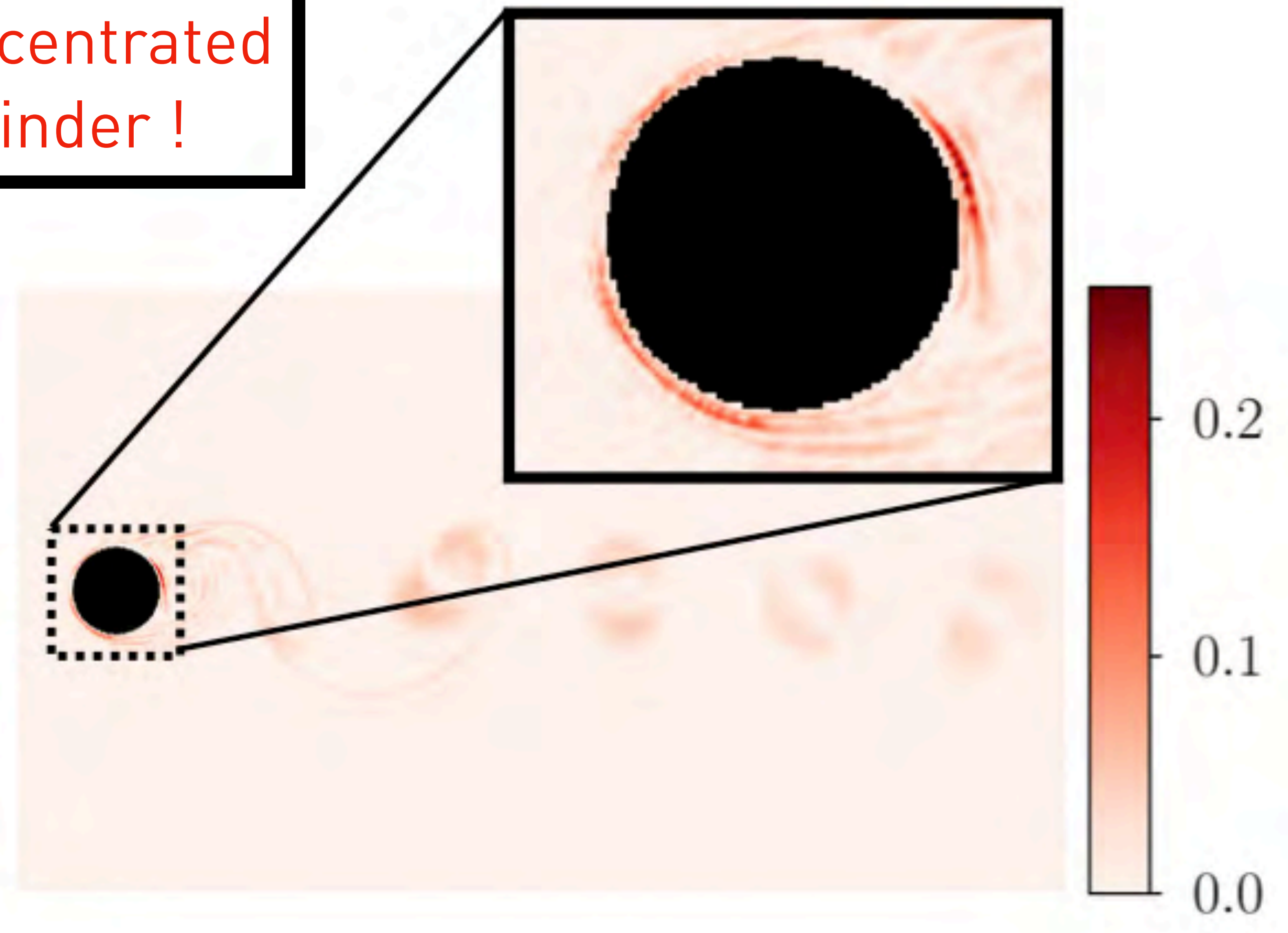


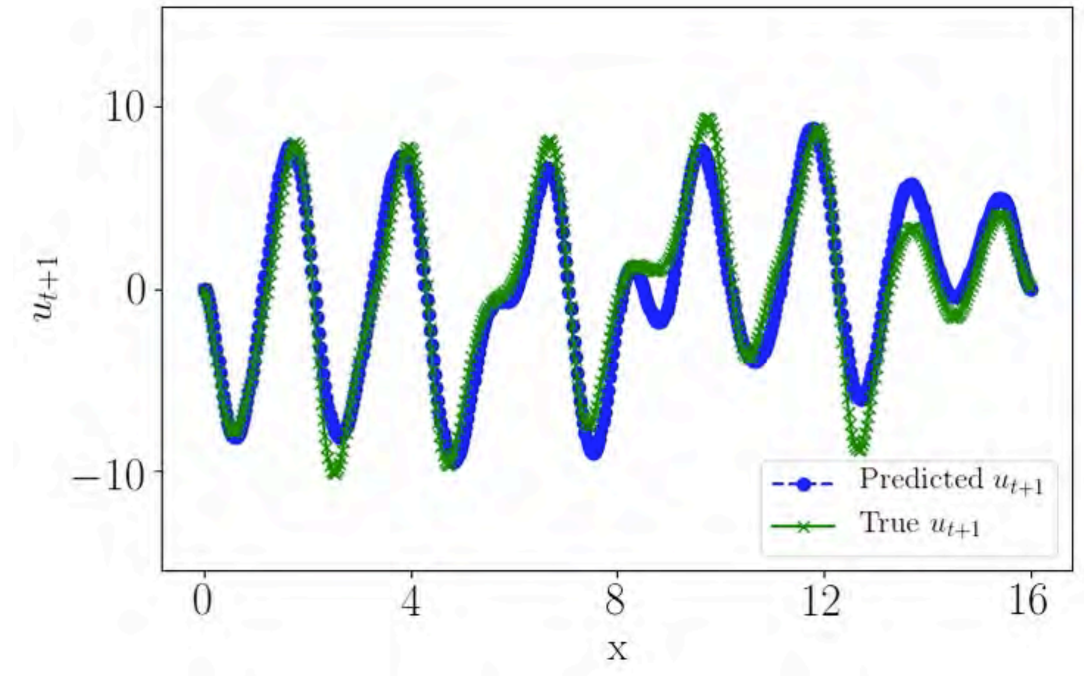


Most error is concentrated around the cylinder !







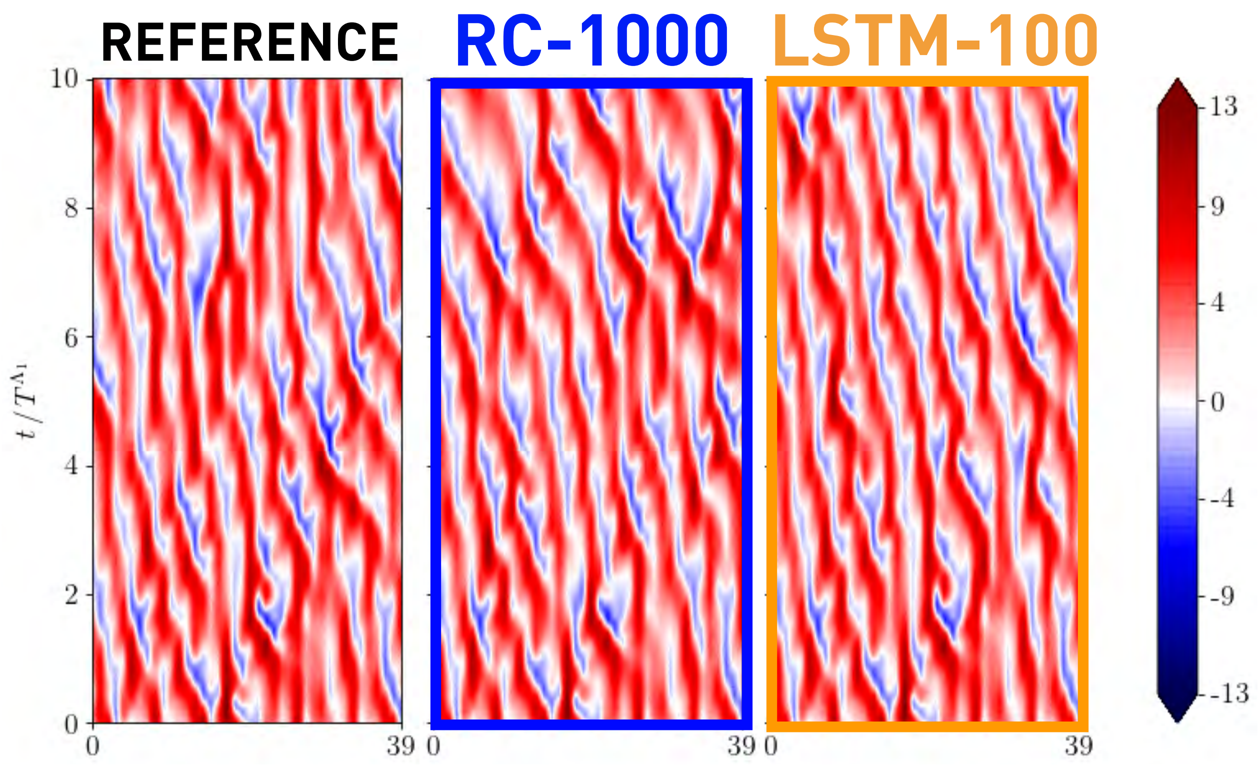


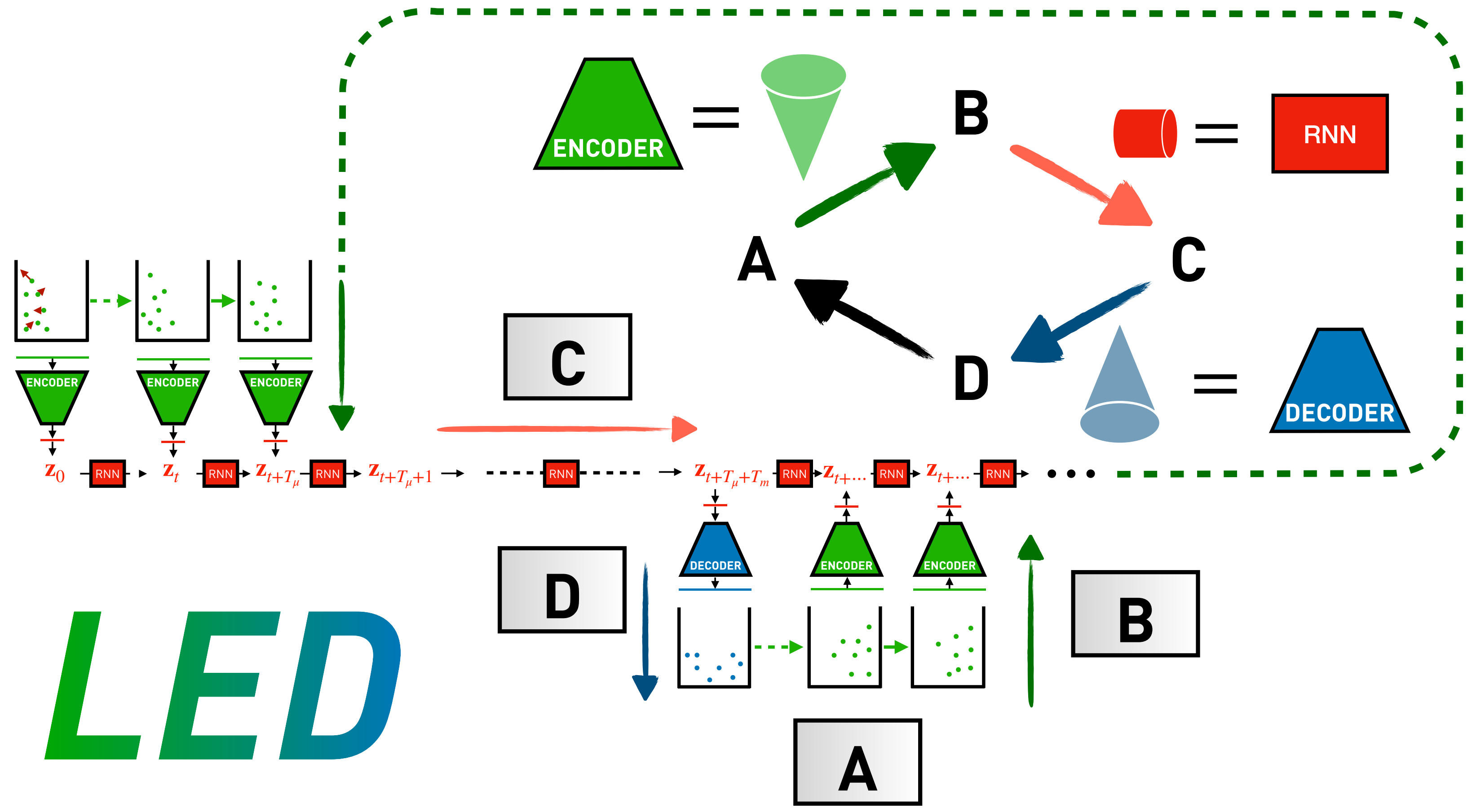
HYBRID LSTM - MSM

$$\dot{z}_t = \begin{cases} \mathrm{LSTM}^{\mathbf{W}}(z_t, z_{t-1}, z_{t-2}, \ldots) & \text{if } p_{train}(z_t) \geq \theta \\ \mathrm{MSM}^{\zeta, c}(z_t) & \text{if } p_{train}(z_t) < \theta \end{cases}$$

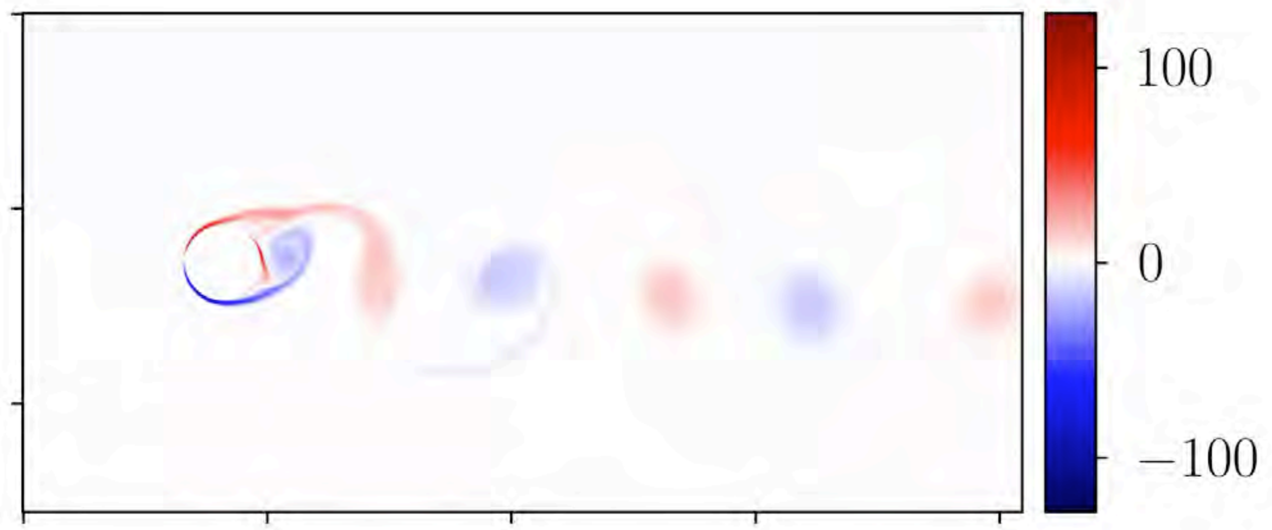
PR Vlachas, W Byeon, Z Wan, T Sapsis, P Koumoutsakos, *Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks,* **Proc. Roy. Soc. A , 2018**

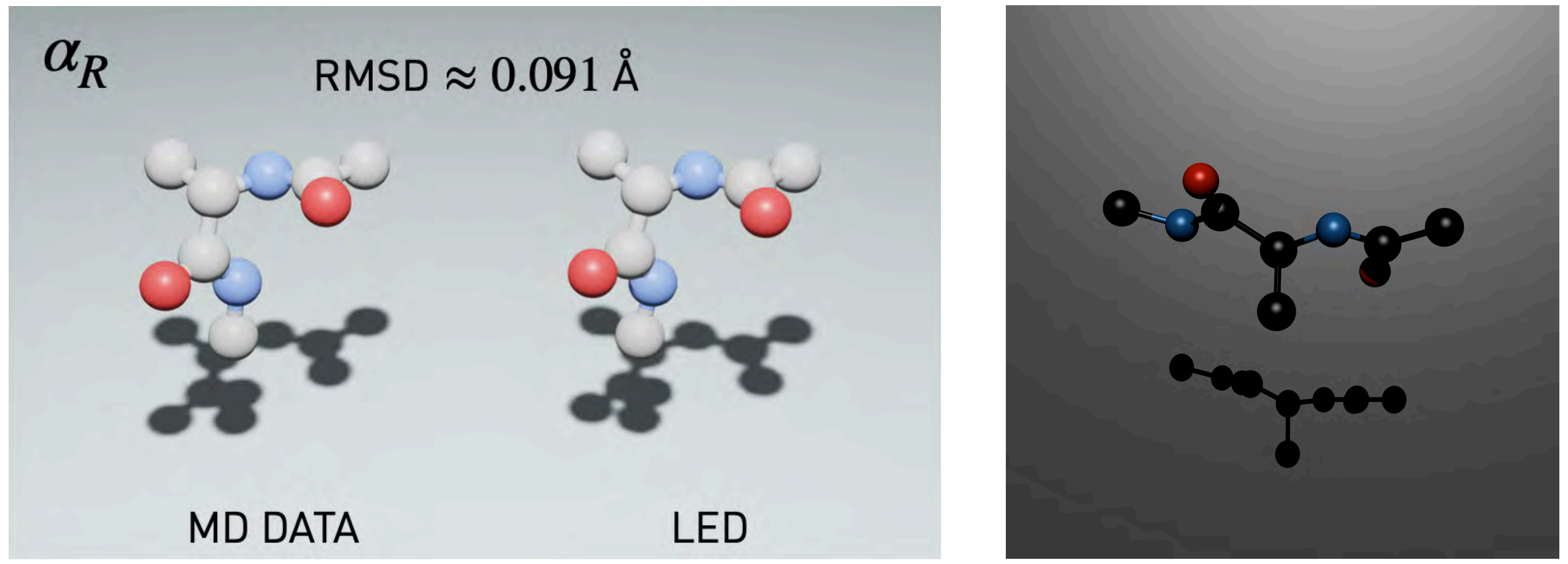
PR Vlachas , J Pathak , BR Hunt , TP Sapsis , M Girvan, E Ott and P Koumoutsakos, *Backpropagation algorithms and Reservoir Computing in Recurrent Neural Networks for the forecasting of complex spatiotemporal dynamics,* **Journal of Neural Networks, 2020**



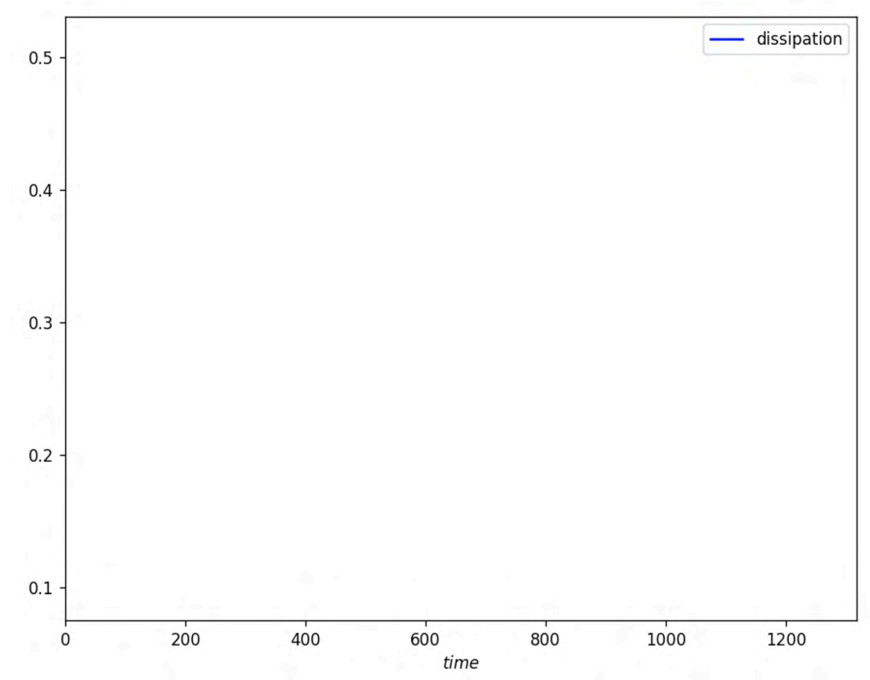


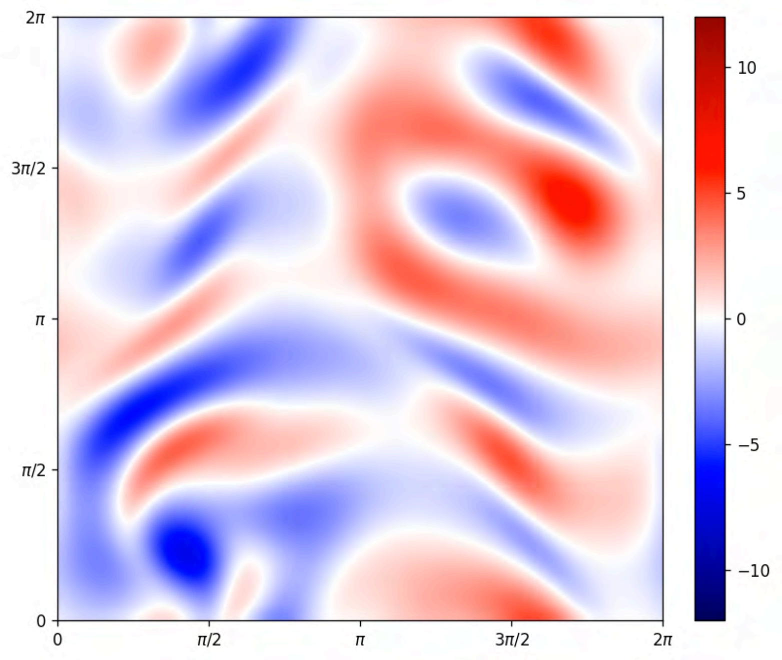
PR Vlachas, G Arampatzis, C Uhler, P Koumoutsakos, *Multiscale Simulations of Complex Systems by Learning their Effective Dynamics,* **Nature Machine Intelligence, 2022**





PR Vlachas, J Zavadlav, M Praprotnik, P Koumoutsakos, *Accelerated Simulations of Molecular Systems through Learning of their Effective Dynamics,* **Journal of Chemical Theory & Computation, 2021**

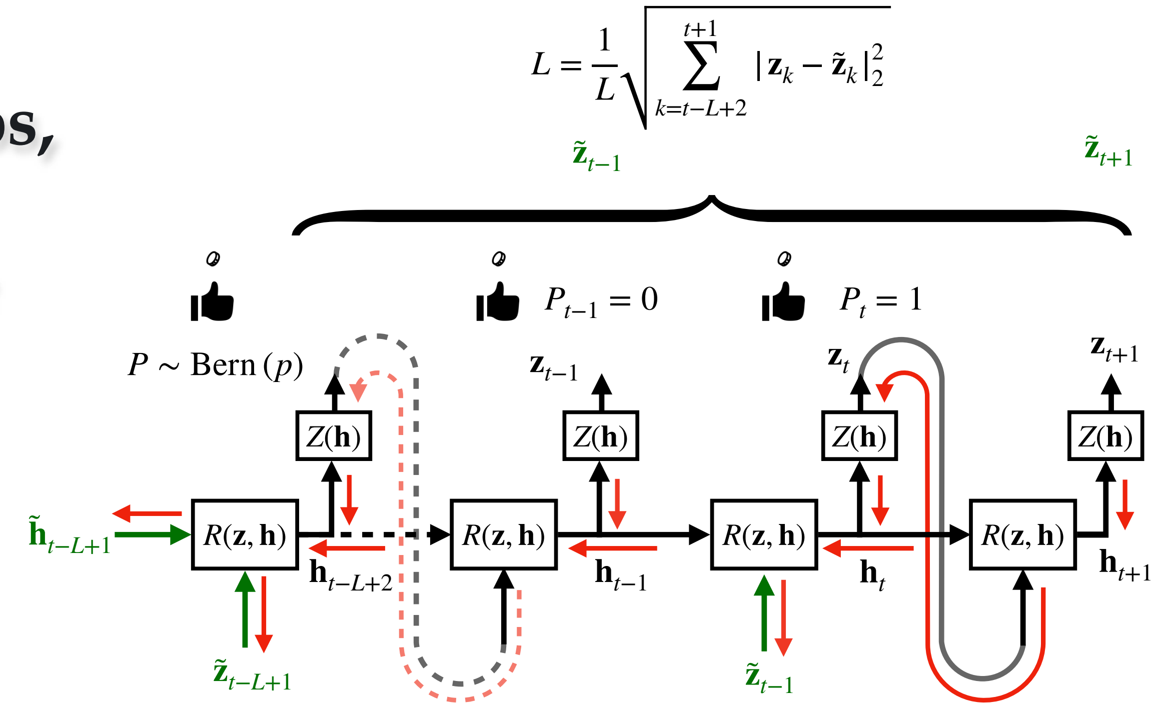




ZY Wan, P Vlachas, P Koumoutsakos, T Sapsis, *Data-assisted reduced-order modeling of extreme events in complex dynamical systems,* **PloS one, 2018**

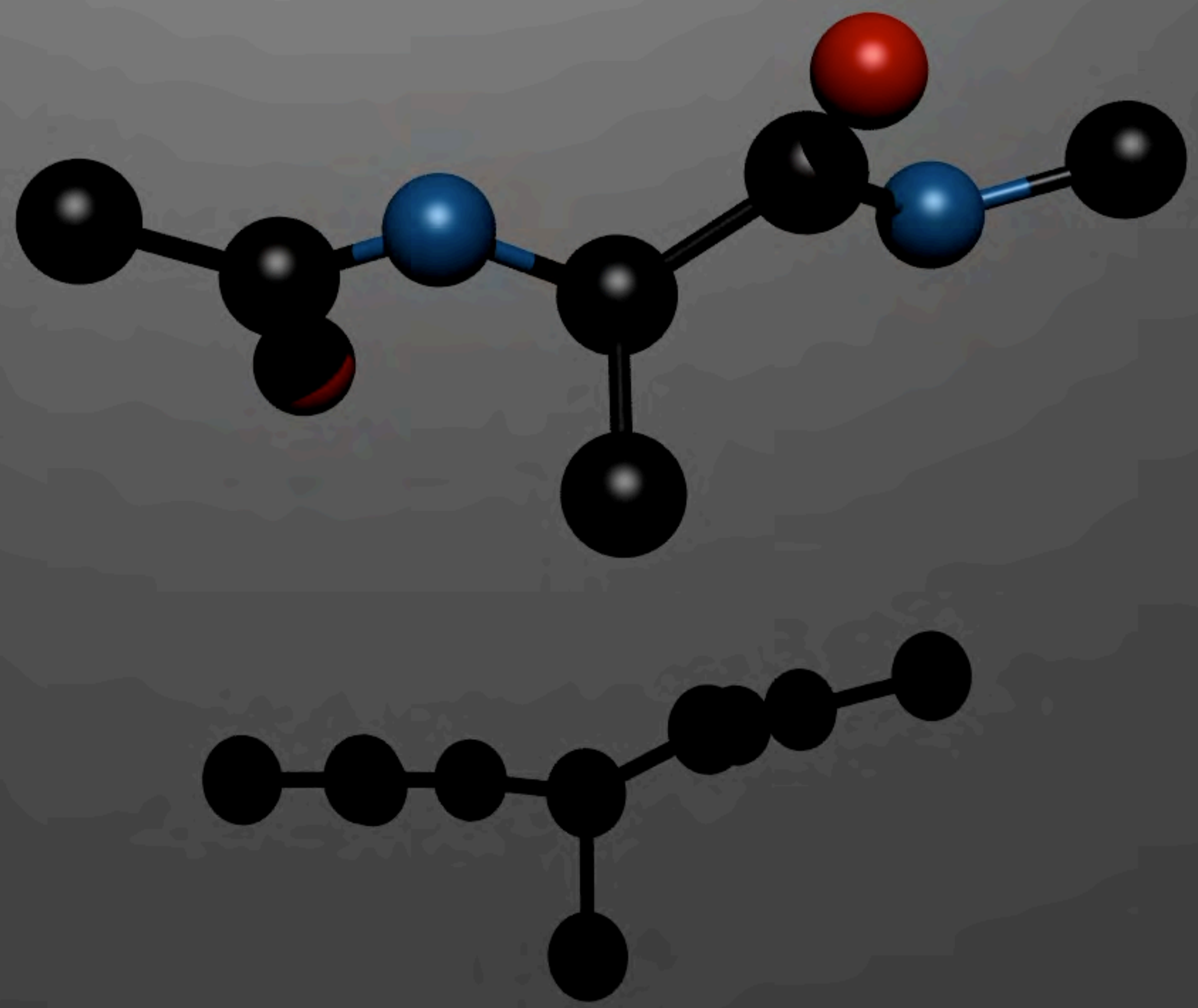
$$\dot{\xi}_t = F(\xi_t) + \tilde{G}(\xi_t, \xi_{t-1}, \xi_{t-2}, \ldots)$$

PR Vlachas, P Koumoutsakos, *Scheduled Autoregressive Backpropagation Through Time for Long-Term Forecasting,* **(in preparation)**

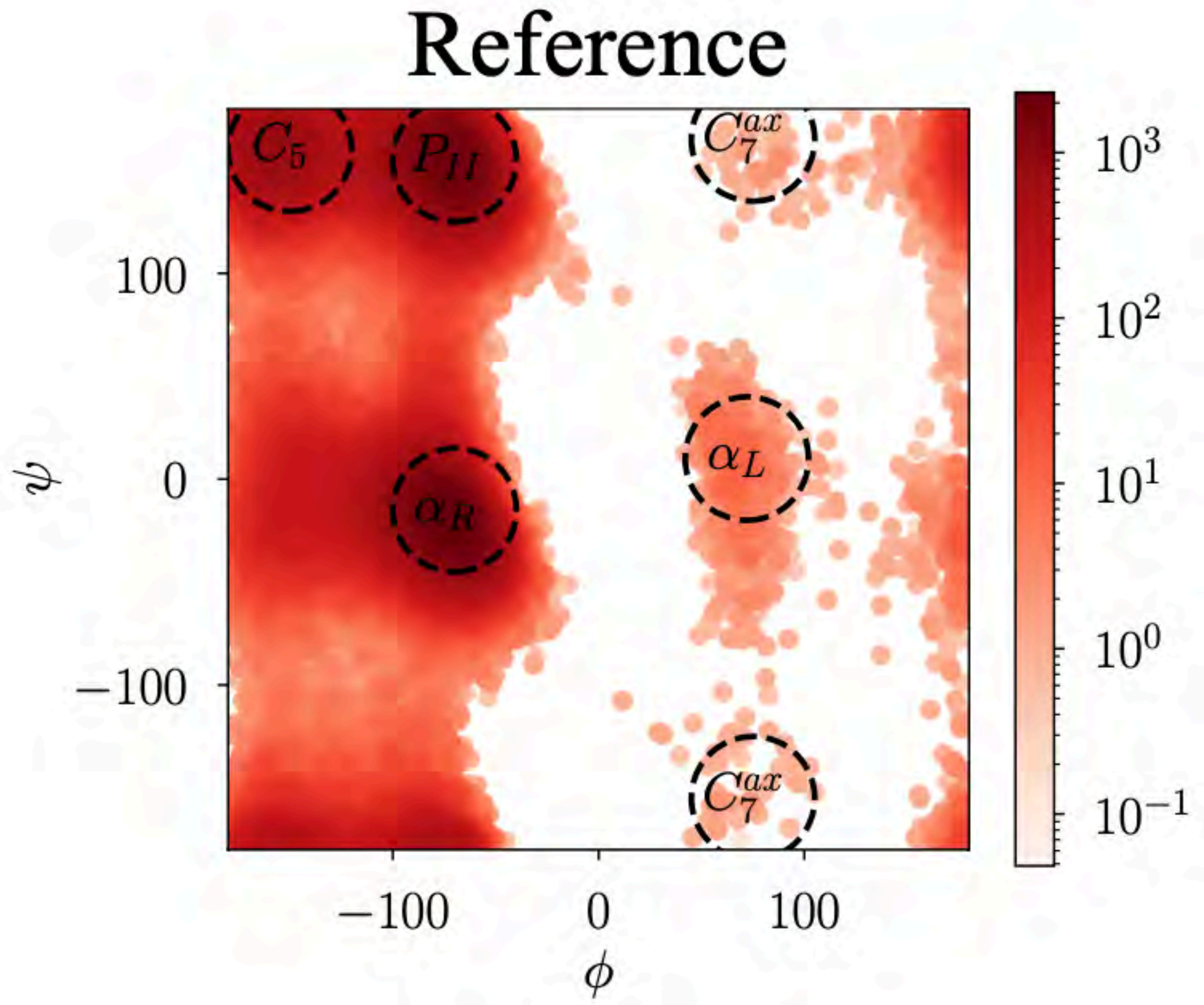


Alanine Dipeptide

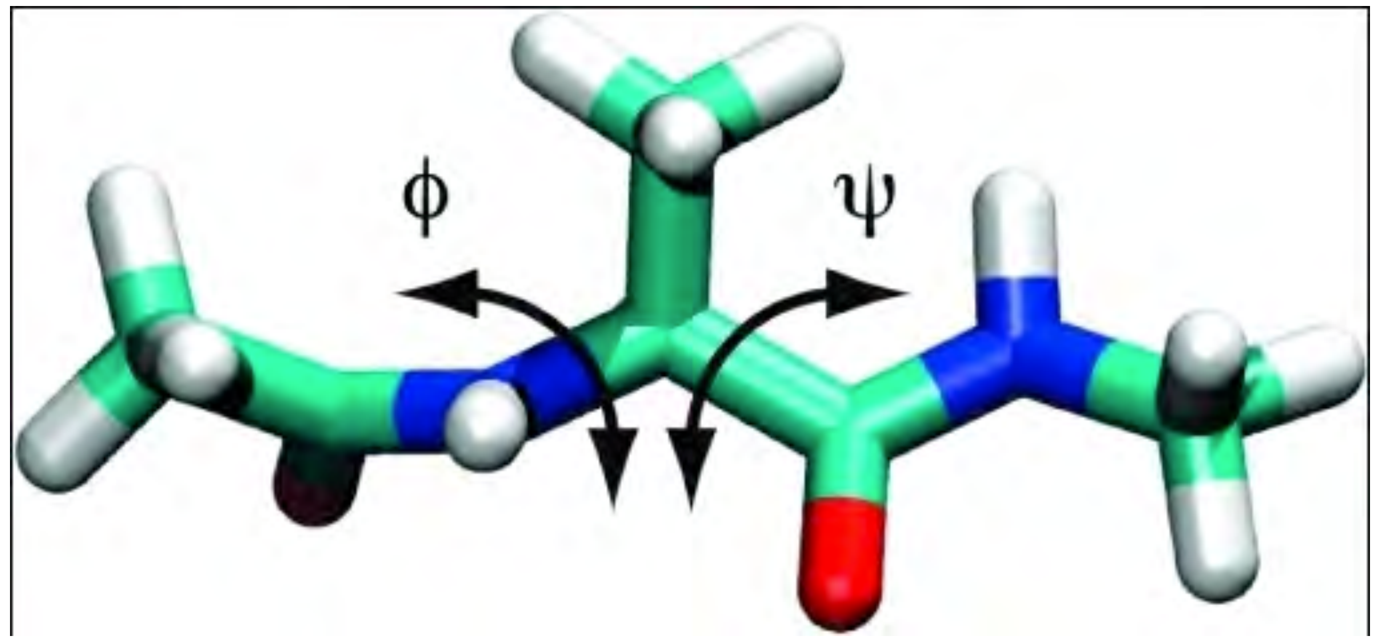
MD simulations (H-not shown)



Stochastic dynamics, transitions between **5 meta-stables** states $\{P_{II}, C_5, \alpha_R, \alpha_L, C_7^{ax}\}$

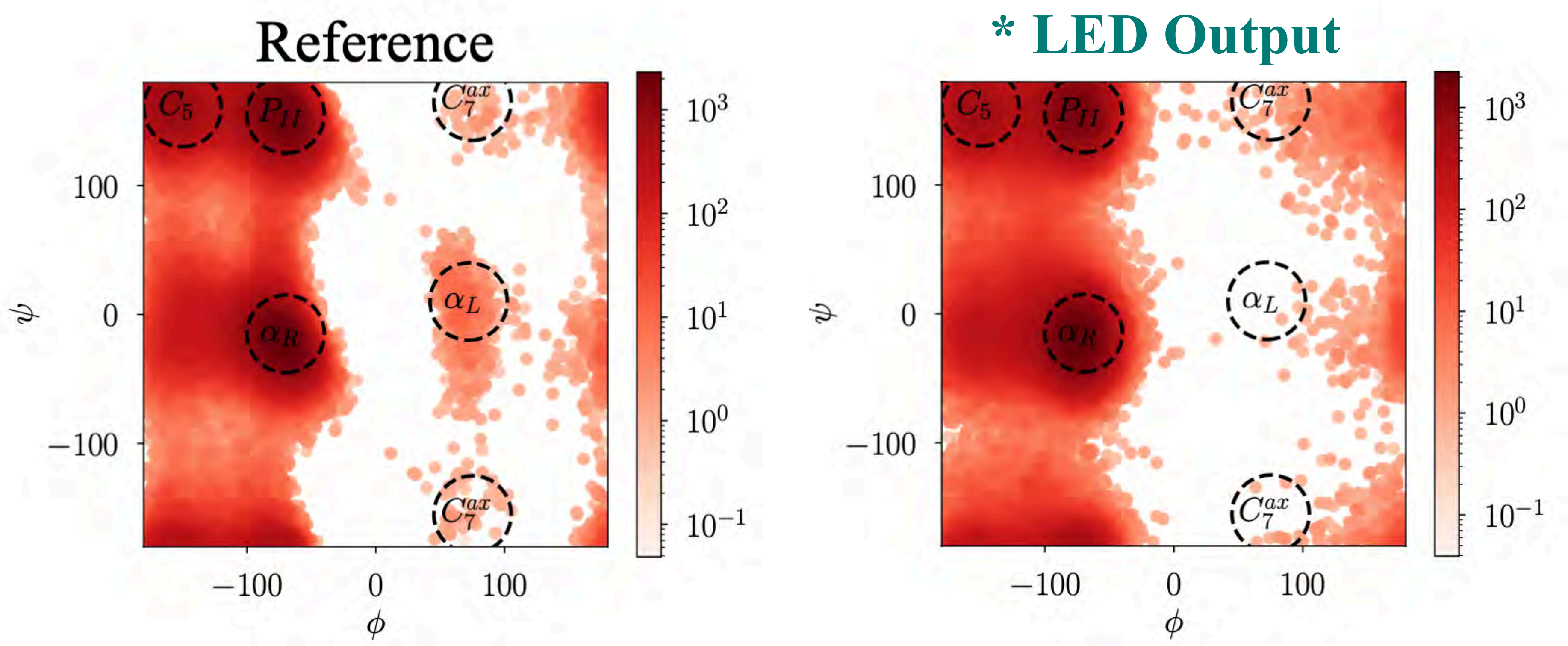


Ramachandran space spanned by (ϕ, ψ)

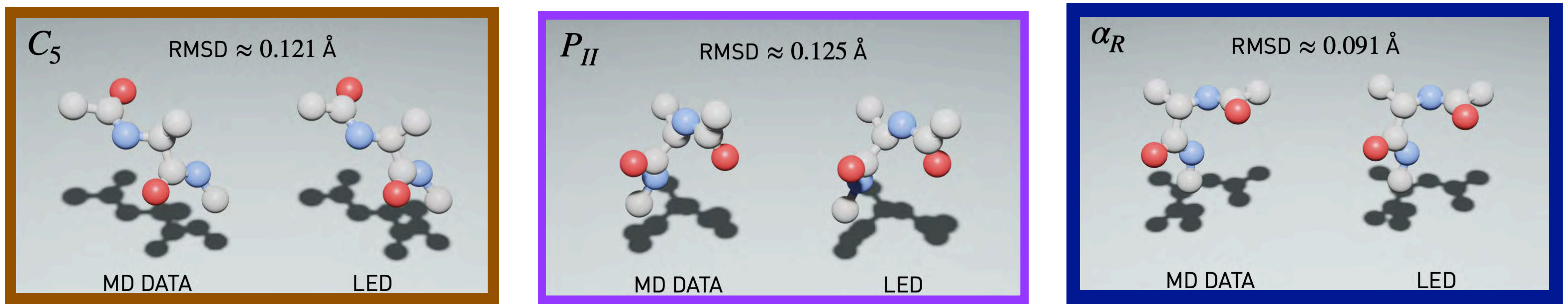


LED in Alanine Dipeptide

PR Vlachas, J Zavadlav, M Praprotnik, P Koumoutsakos,
Accelerated Simulations of Molecular Systems through Learning of their Effective Dynamics,
Journal of Chemical Theory & Computation, 2021

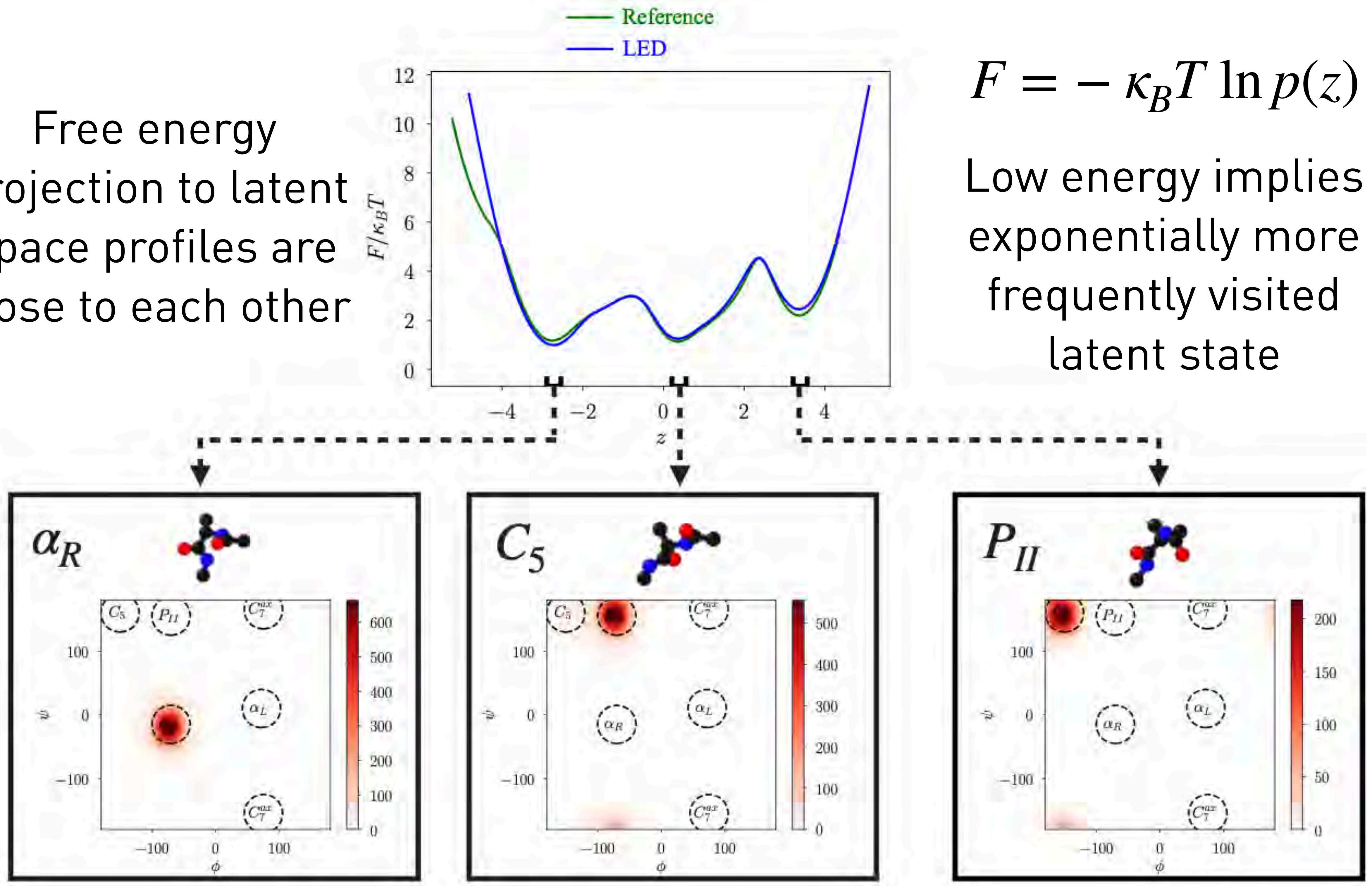


Ramachandran space spanned by (ϕ, ψ)



- Model only (no water, hydrogen etc.) heavy atoms $\mathbf{s}_t \in \mathbb{R}^{24}$, (angles, bond lengths, dihedral angles) and **LED with** $d_{\mathbf{z}} = 1$
- LED:
  - reproduces coarsely the statistics
  - reproduces kinetics, and mean transition times (see paper)
  - samples realistic configurations
  - maps different regions in the latent space, to known configurational states
- LED is **three orders faster** than MD

Free energy projection to latent space profiles are close to each other



$$F = -\kappa_B T \ln p(z)$$

Low energy implies exponentially more frequently visited latent state

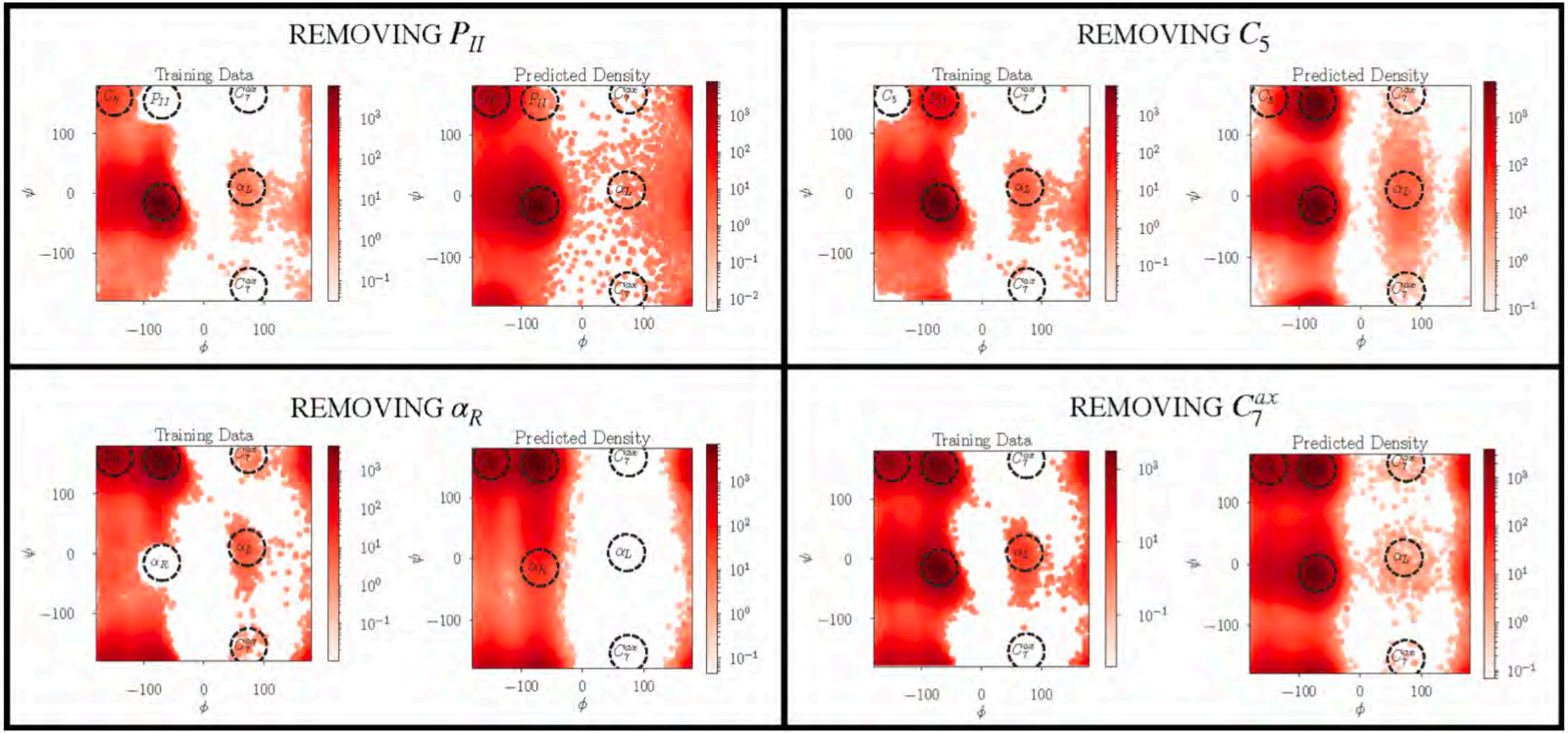


Figure 9. LED is trained in four scenarios hiding data that lie closer than 40° to one of the metastable states $\{P_{\mathrm{II}}, C_5, \alpha_{\mathrm{R}}, C_7^{\mathrm{ax}}\}$ each time. LED can successfully generate novel probable configurations close to the metastable states $\{P_{\mathrm{II}}, C_5, \alpha_{\mathrm{R}}, C_7^{\mathrm{ax}}\}$. Due to the limited training data, however, capturing the state density in the Ramachandran plot is challenging.

LECTURES

- **LECTURE 1:**
  - What is Artificial Intelligence? How it relates to Computing ?
  - How are these relevant to modeling, understanding, optimizing complex systems?

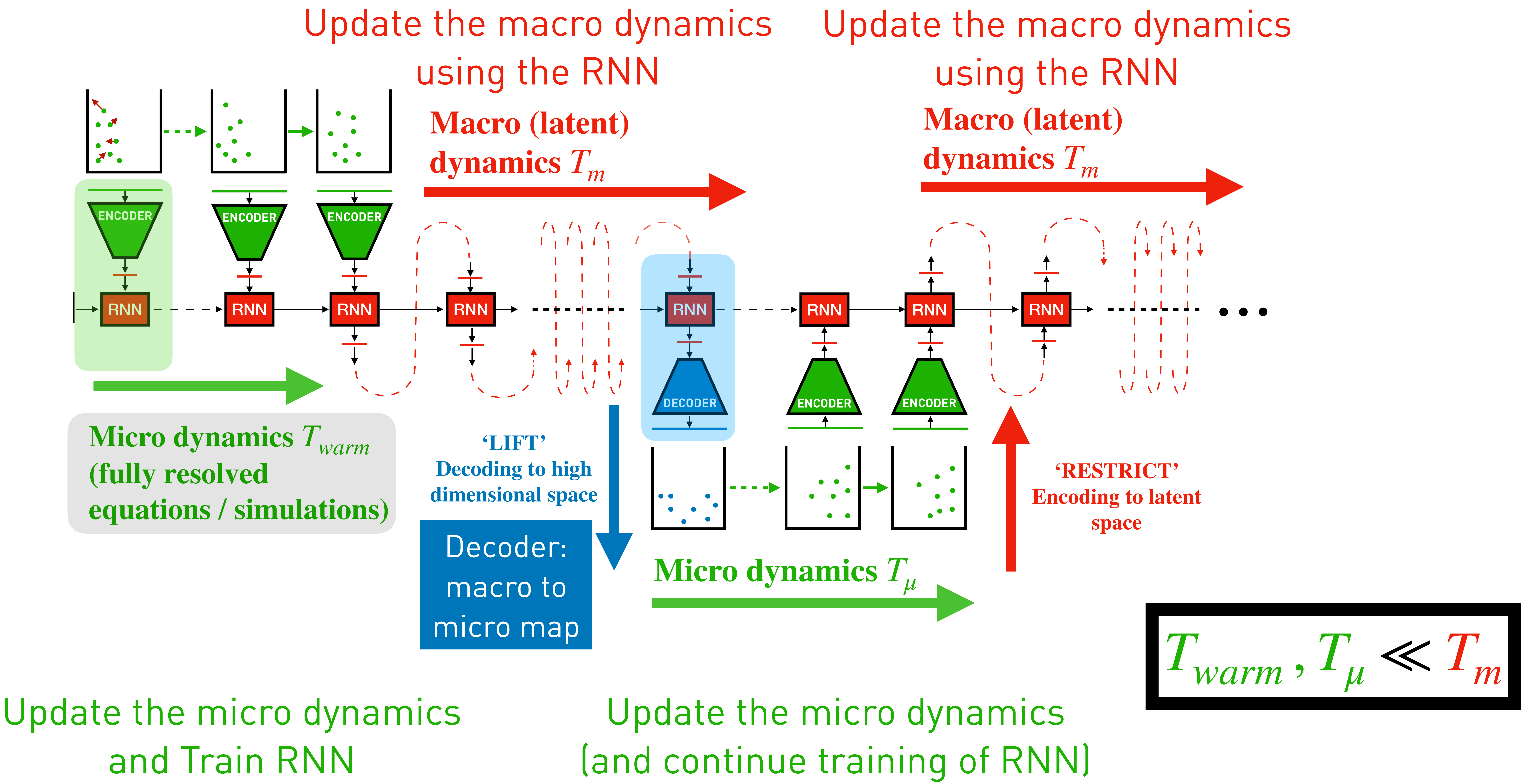
- **LECTURE 2:**
  - (Machine) Learning of Effective Dynamics of Complex Systems.
  - Reinforcement Learning for Controlling Complex Systems.

- **LECTURE 3**
  - Generative LED
  - Learning to solve PDEs without Machine Learning
  - Learning to solve PDEs with Reinforcement Learning Machine

- **LECTURE 4**
  - Uncertainty Quantification & Stochastic Optimization
  - Optimal Sensor Placement

Learning Effective Dynamics

PR Vlachas, G Arampatzis, C Uhler, P Koumoutsakos, *Multiscale Simulations of Complex Systems by Learning their Effective Dynamics*, Nature Machine Intelligence, (2022)



Adaptive LED

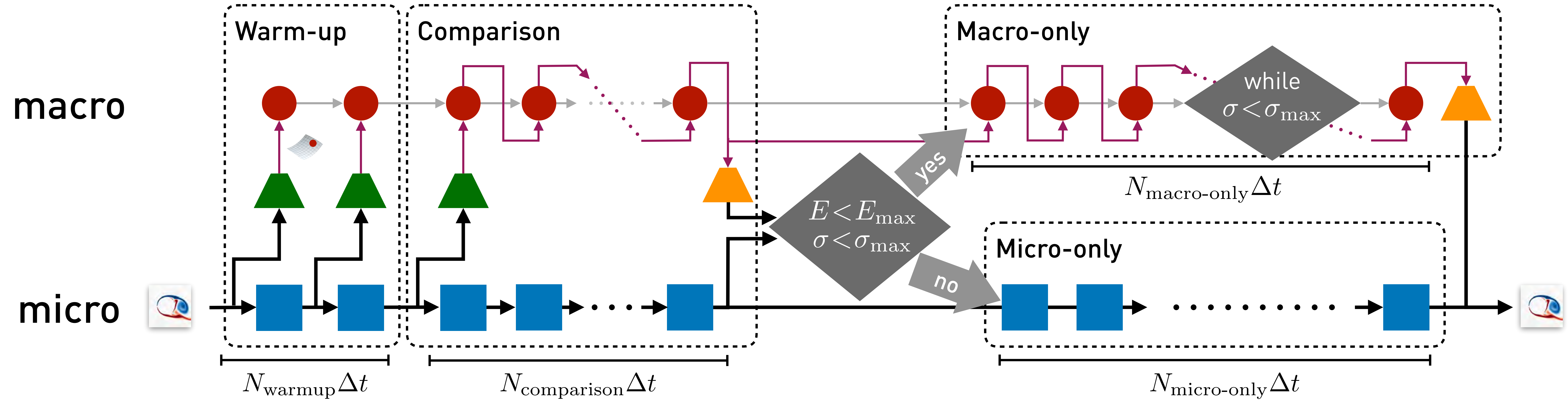
Computer Methods in Applied Mechanics and Engineering
Volume 415, 1 October 2023, 116204

Adaptive learning of effective dynamics for online modeling of complex systems

Ivica Kičić [a], Pantelis R. Vlachas [a b], Georgios Arampatzis [a b], Michail Chatzimanolakis [a b], Leonidas Guibas [c], Petros Koumoutsakos [b]

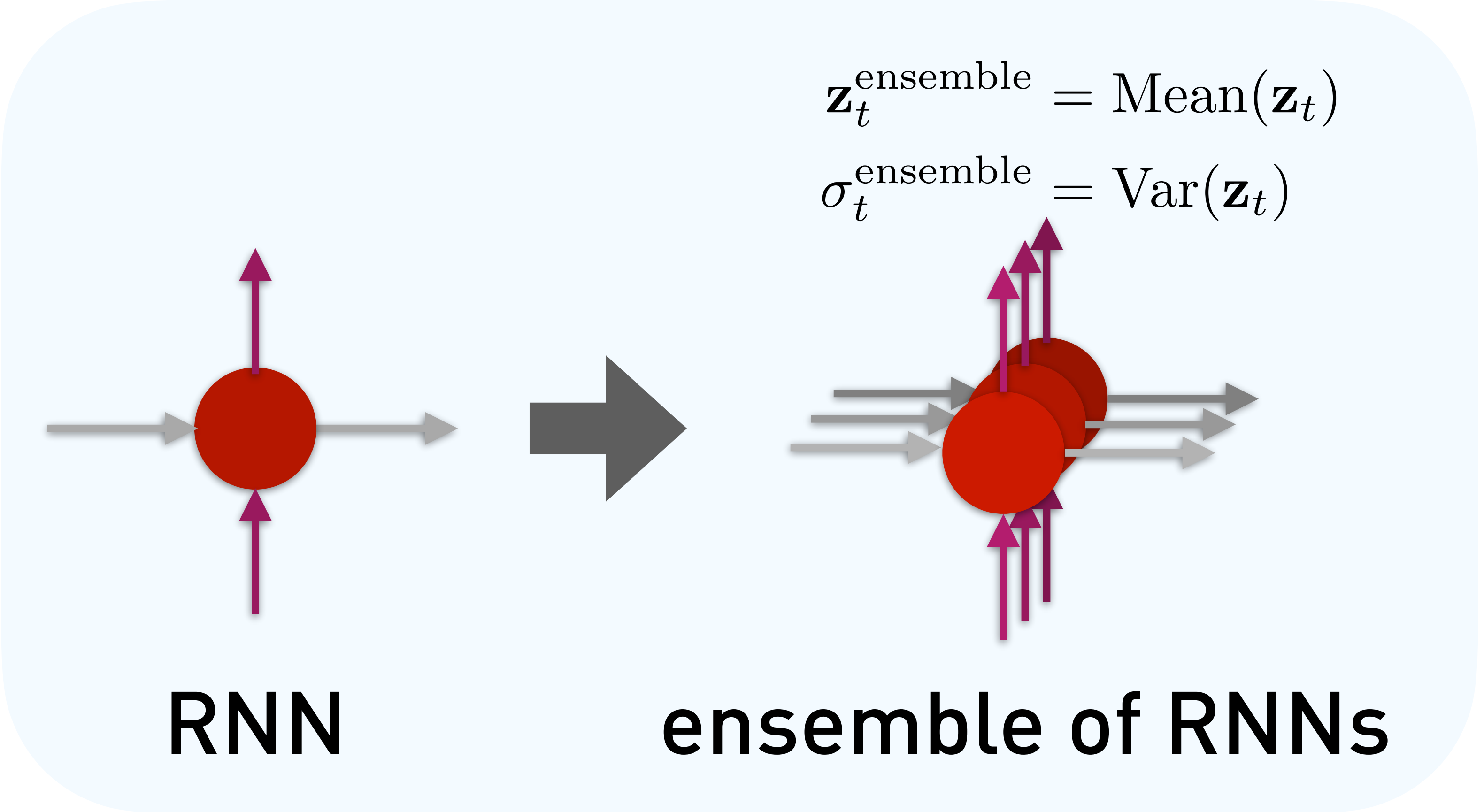
AdaLED cycle (inference)

Kičić, Vlachas, Arampatzis, Chatzimanolakis, Guibas, Koumoutsakos , CMAME, 2023



Flow behind the cylinder:

- 1000ms (20 CFD time steps) (1/50 of a period)
- 5ms
- 50ms
- 50ms



Lakshminarayanan et al. (2016)

Legend:

- $\mathbf{x}_t$ micro state
- $\mathbf{z}_t$ macro state
- $\mathbf{h}_t$ hidden state
- $E_{\max}$ error threshold
- $\sigma_{\max}$ uncertainty threshold

- micro propagator (solver)
- macro propagator (RNNs)
- encoder
- decoder

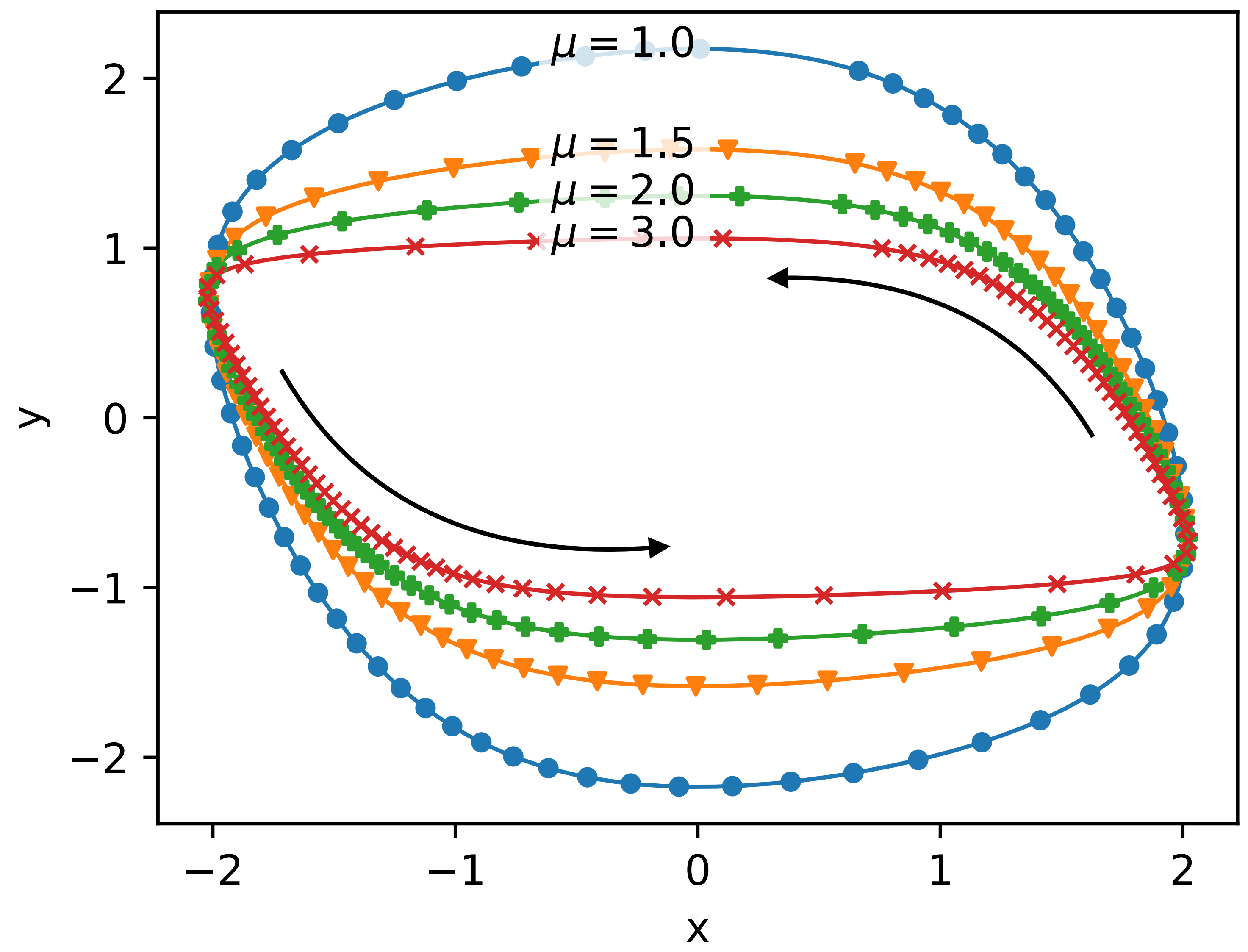
Case study: Van der Pol oscillator

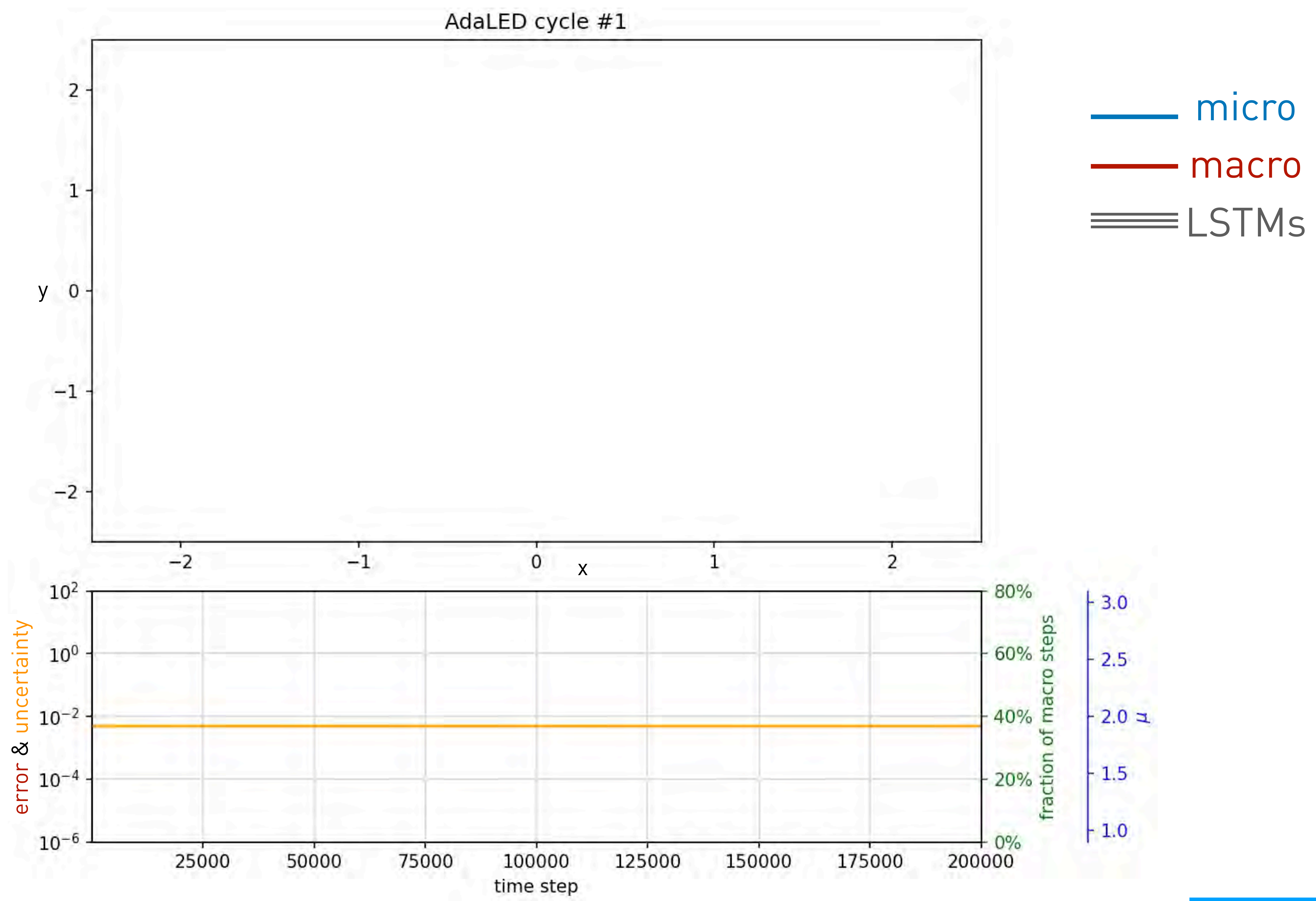
Non-linear ODE system:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \mu\left(x - \frac{1}{3}x^3 - y\right)$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \frac{1}{\mu}x$$

Time-varying system parameter





3X fewer μ-steps

Generative LED

arXiv > cs > arXiv:2402.17157

Search...

Help | Advar

Computer Science > Machine Learning

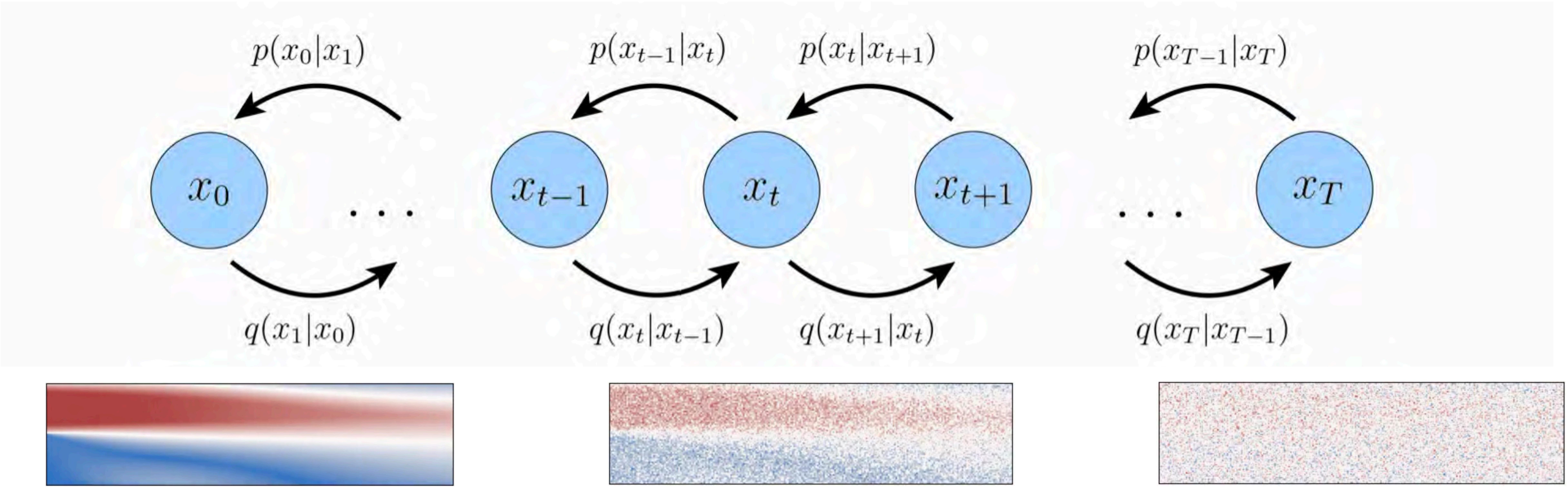
[Submitted on 27 Feb 2024]

Generative Learning for Forecasting the Dynamics of Complex Systems

Han Gao, Sebastian Kaltenbach, Petros Koumoutsakos

Visual representation of a Variational Diffusion Model.

2. A diffusion model learns to reverse this process.

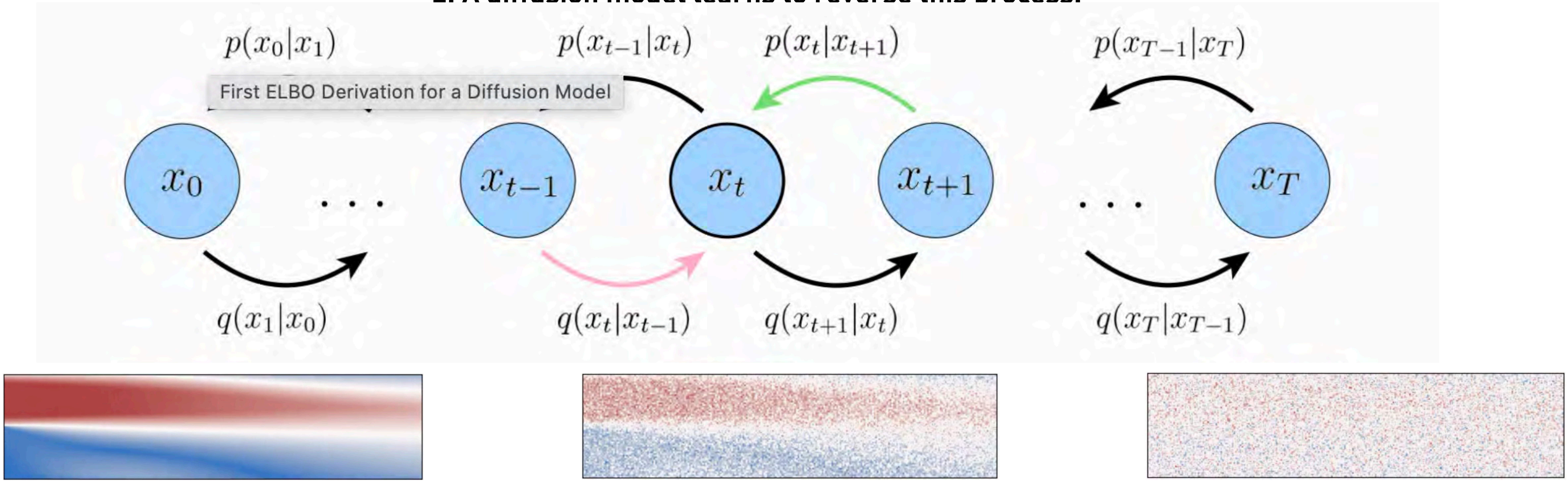


1. An input is steadily noised over time until it becomes identical to Gaussian noise

https://calvinyluo.com/2022/08/26/diffusion-tutorial.html

Visual representation of a Variational Diffusion Model (VDM)

2. A diffusion model learns to reverse this process.



1. An input is steadily noised over time until it becomes identical to Gaussian noise

A VDM can be optimized by ensuring that for every intermediate latent, the posterior from the latent above it matches the Gaussian corruption of the latent before it.

Here, for each intermediate latent, we minimize the difference between the distributions represented by the pink and green arrows.

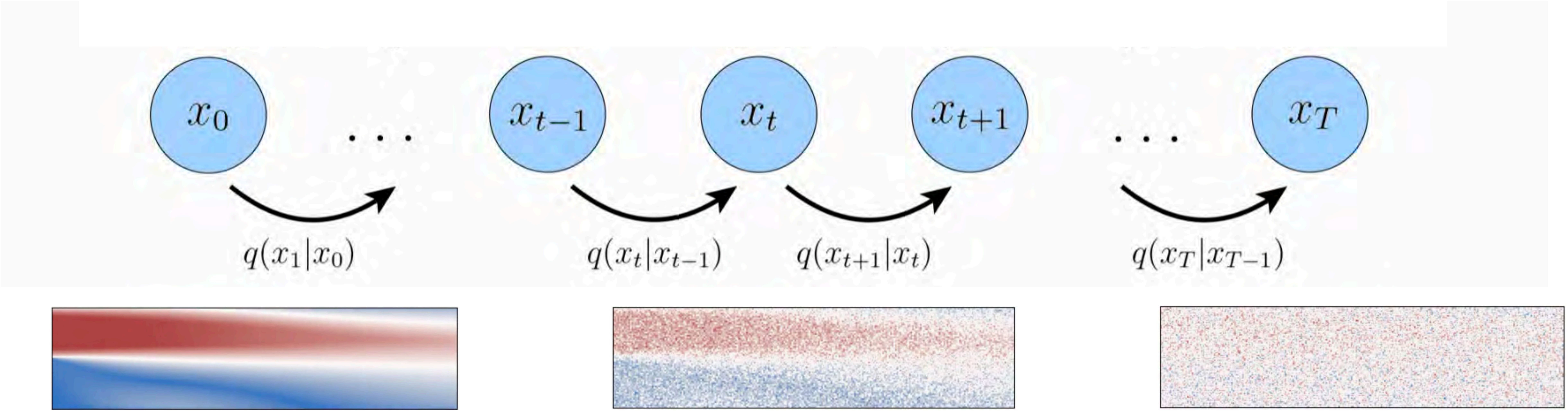
GUIDANCE

Generate data from a conditional distribution $p(x \mid z)$ through **conditioning information** z .

IDEA: Use the Latent Dynamics as Guidance

Backbone of image super-resolution models such as Cascaded Diffusion Models, as well as state-of-the-art image-text models such as DALL-E 2 and Imagen https://calvinyluo.com/2022/08/26/diffusion-tutorial.html

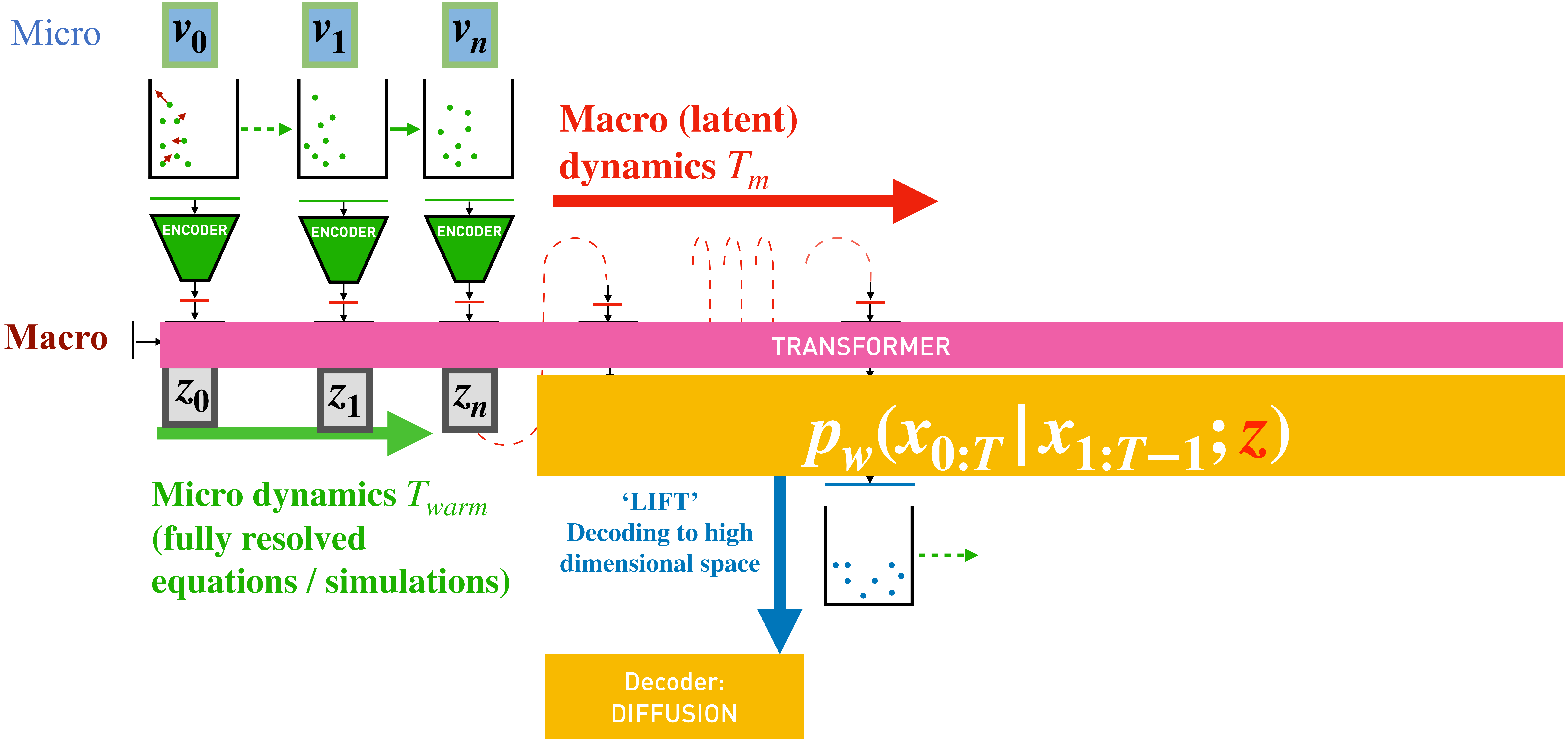
GUIDED Variational Diffusion Model .



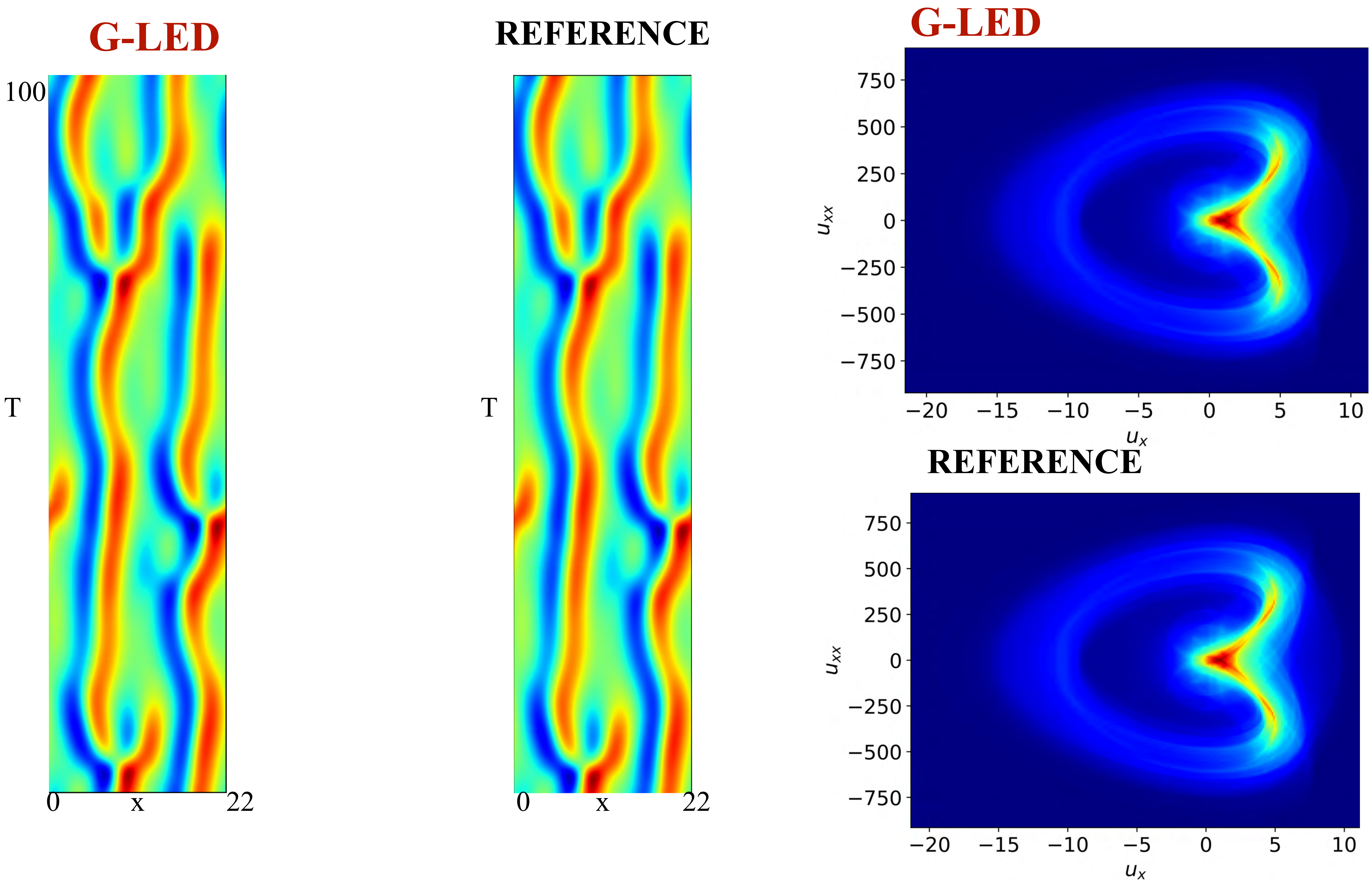
1. Input is steadily noised until it becomes identical to Gaussian noise

TRAINING
Diffusion Models
v_1
Micro level
v_2
… … … …
v_n
$x_0 = (v_1, v_2, \ldots, v_n)$
$z = (z_1, z_2, \ldots, z_n)$
Physics Information
$p_w(x_0 \mid x_1; z)$
$p_w(x_{t-1} \mid x_t; z)$
$p_w(x_t \mid x_{t+1}; z)$
$p_w(x_{T-1} \mid x_T; z)$
Guided diffusion model
z_0
Multi-head, auto-regressive attention $(\mathcal{Q}, \mathcal{K}, \mathcal{V})$ model
z_1
Multi-head, auto-regressive attention $(\mathcal{Q}, \mathcal{K}, \mathcal{V})$ model
z_2
…
Multi-head, auto-regressive attention $(\mathcal{Q}, \mathcal{K}, \mathcal{V})$ model
z_n
Macro level
Downsampler
v_0
Micro level

Generative Learning Effective Dynamics



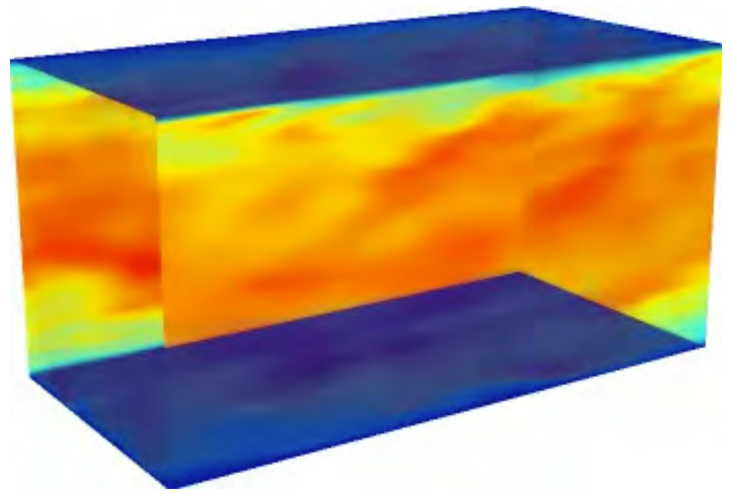
KS Equation



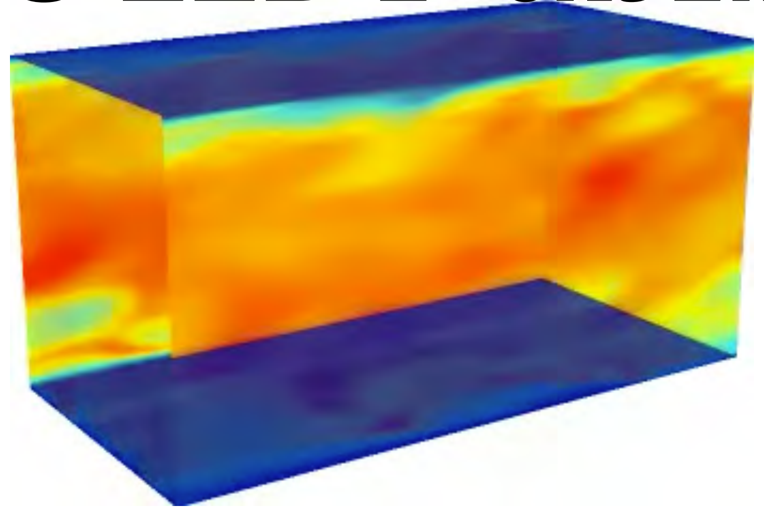
G-LED (left) on a test trajectory with new initial condition.

Turbulent channel flow $Re_\tau = 395$

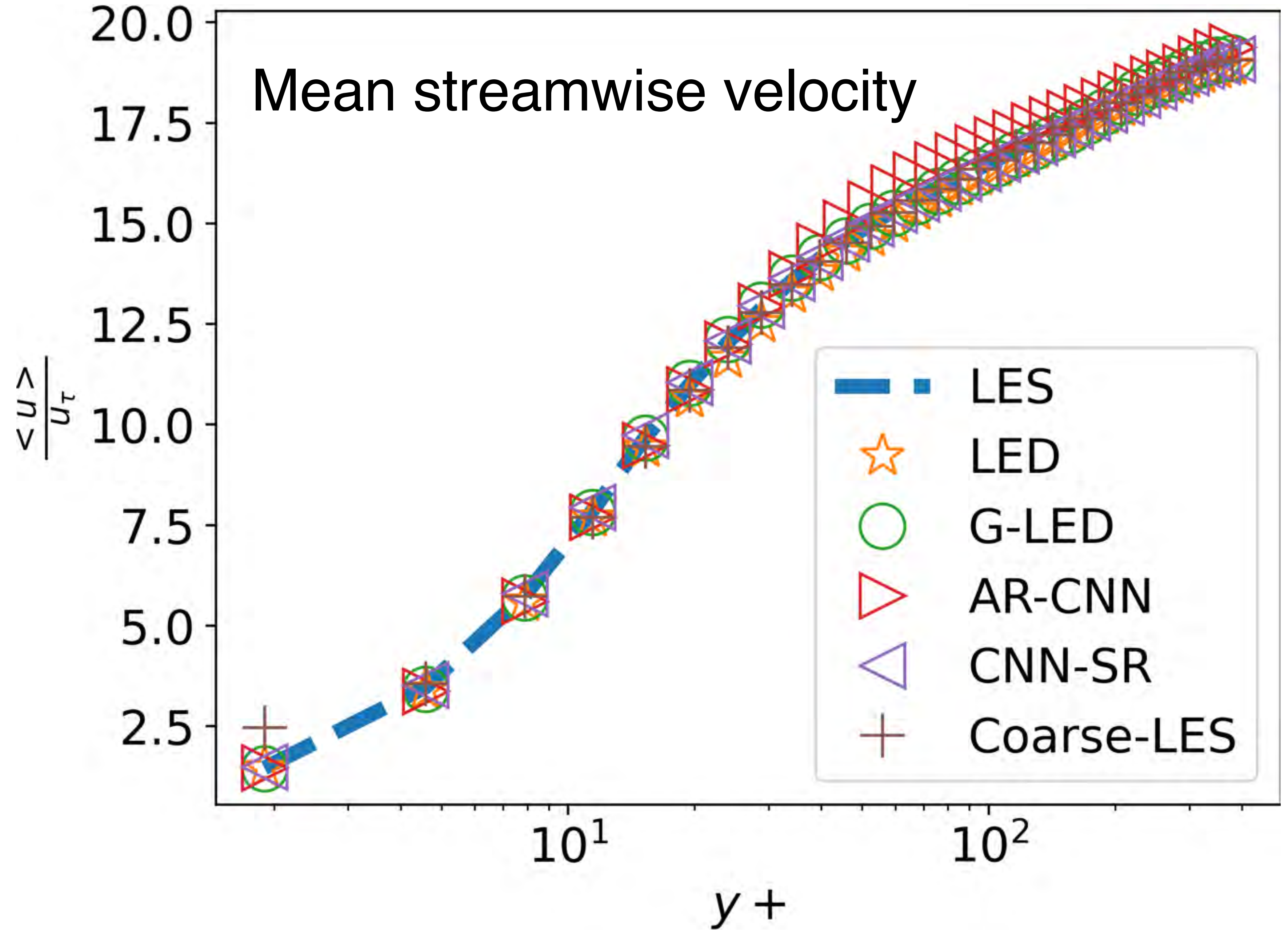
LES = 40x50x30



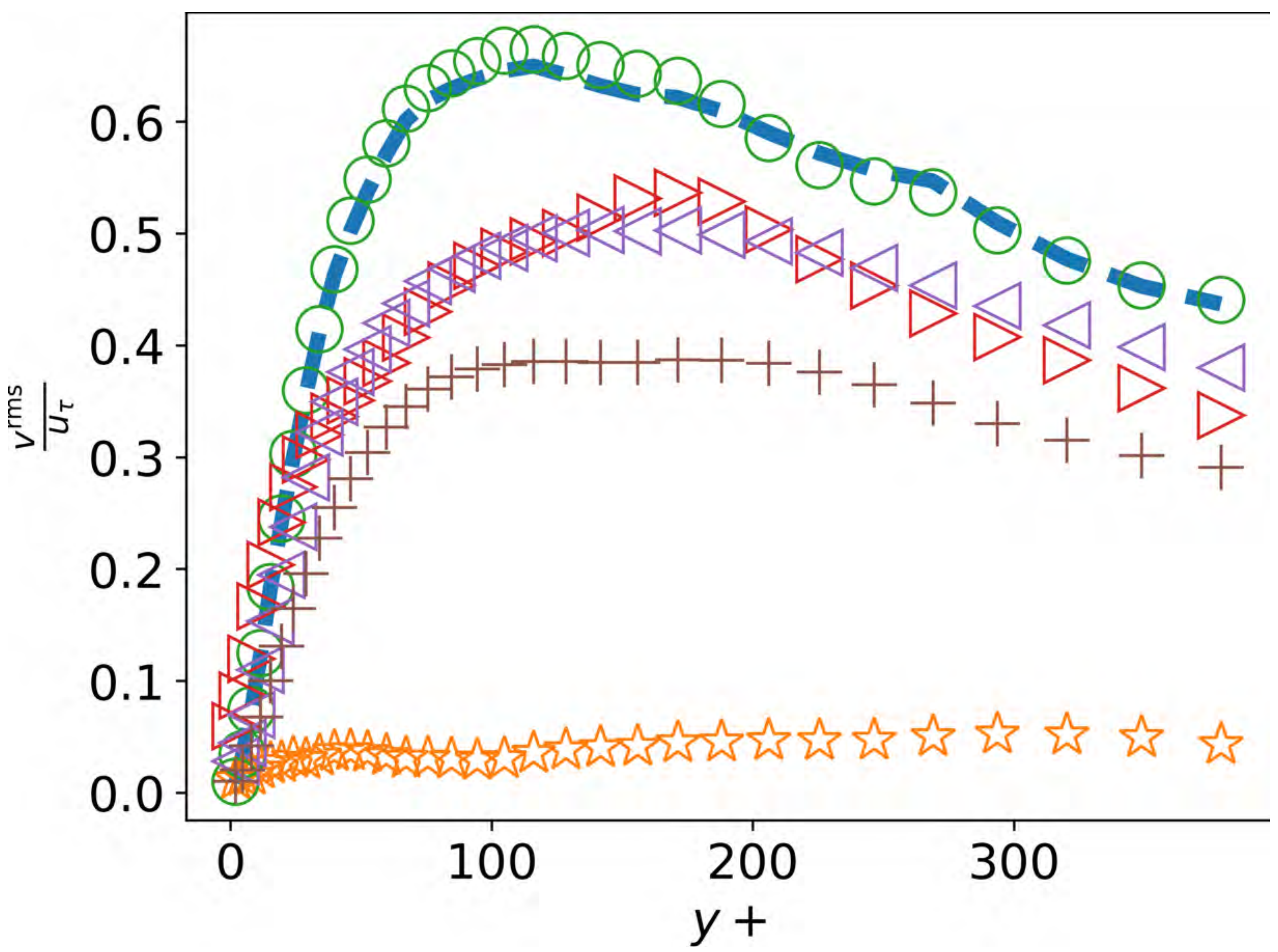
G-LED z=8x32x8



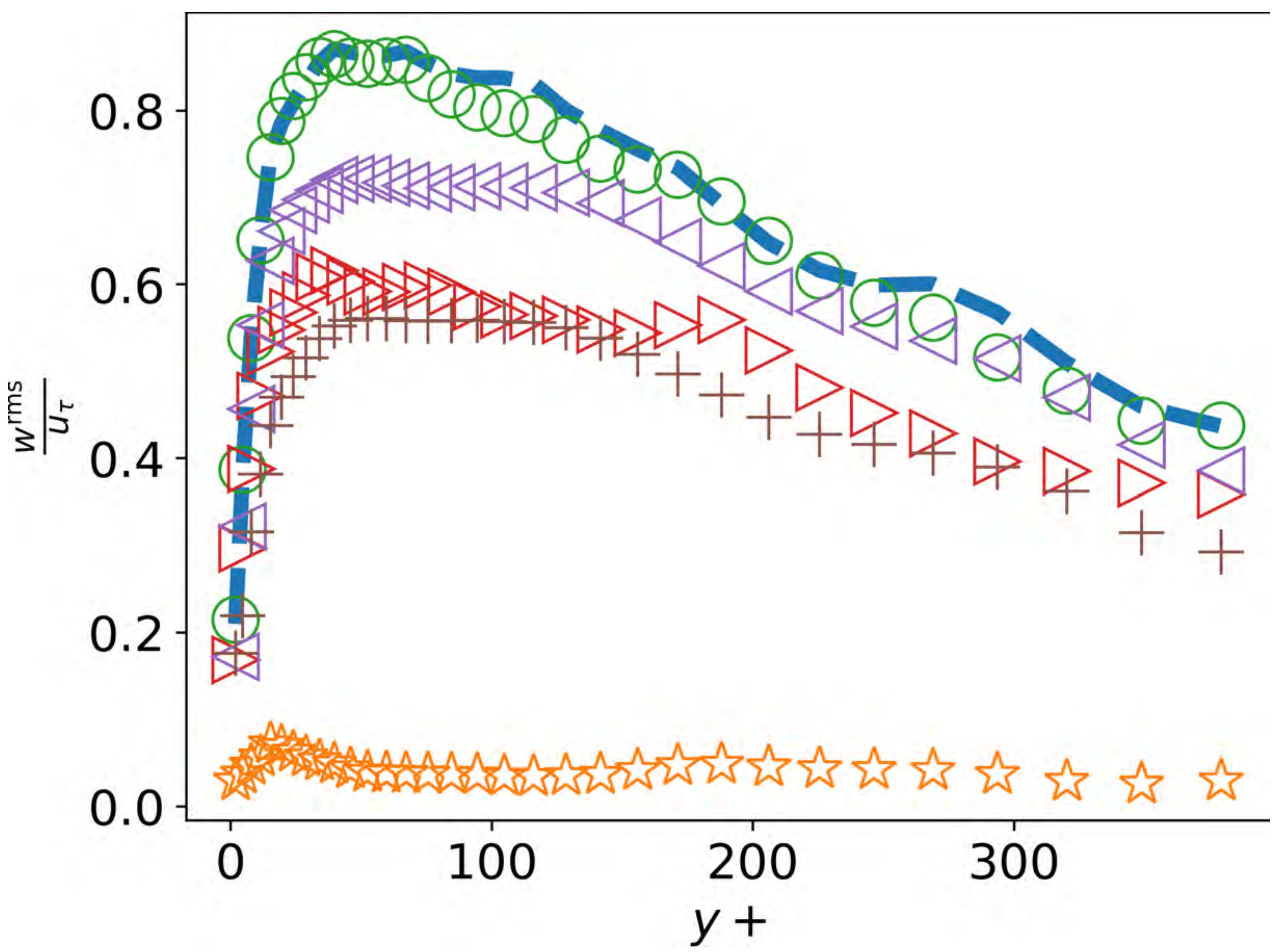
LED	Vlachas, P. R., Arampatzis, G., Uhler, C., & Koumoutsakos, P. (2022). Multiscale simulations of complex systems by learning their effective dynamics. *Nature Machine Intelligence*, *4*(4), 359-366.
LES	Nicoud, F., & Ducros, F. (1999). Subgrid-scale stress modelling based on the square of the velocity gradient tensor. *Flow, turbulence and Combustion*, *62*(3), 183-200.
AR-CNN	Geneva, N., & Zabaras, N. (2020). Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks. *Journal of Computational Physics*, *403*, 109056.
CNN-SR	Ren, P., Rao, C., Liu, Y., Ma, Z., Wang, Q., Wang, J. X., & Sun, H. (2023). PhySR: Physics-informed deep super-resolution for spatiotemporal data. *Journal of Computational Physics*, *492*, 112438.



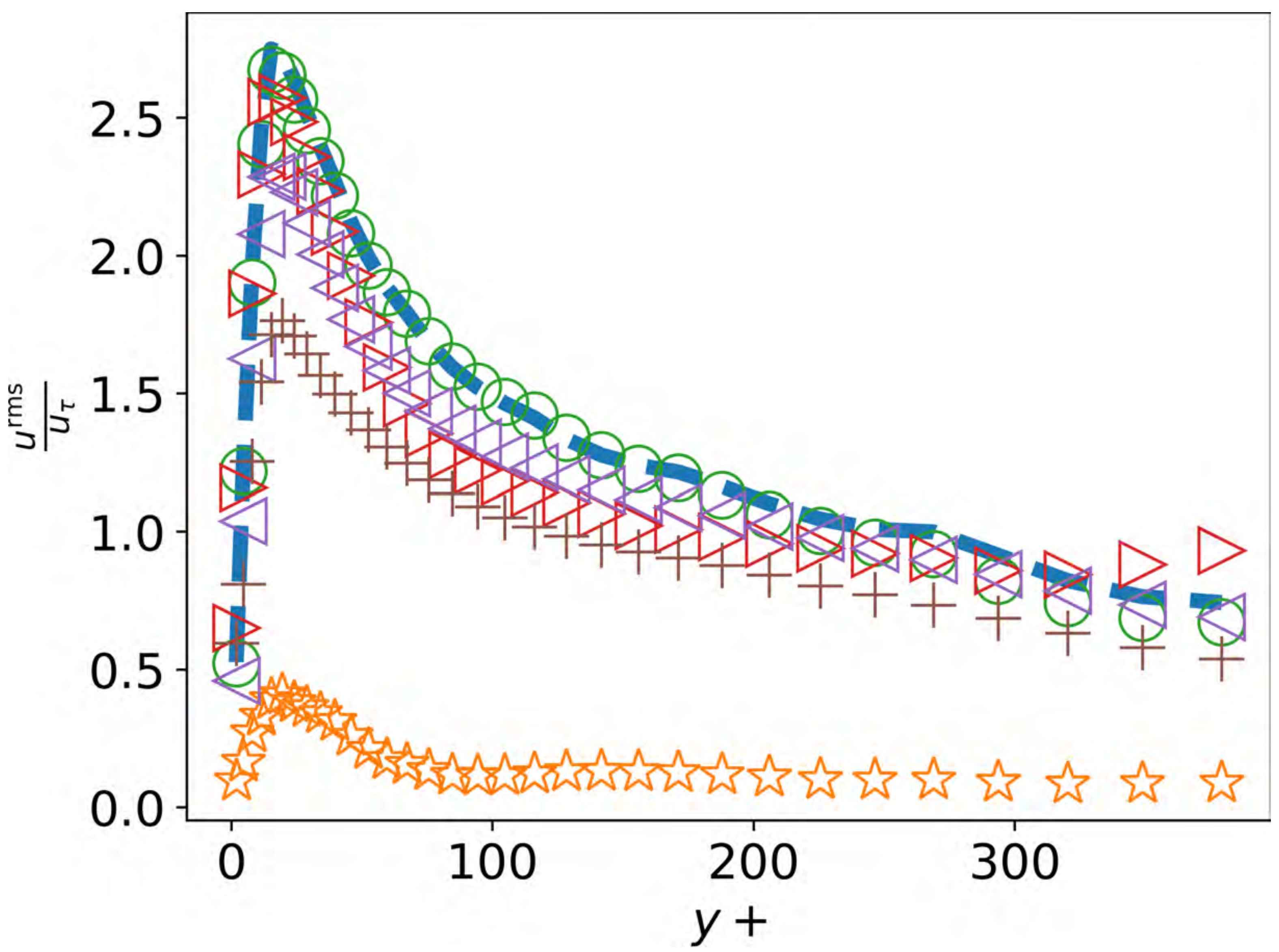
wall normal fluctuations,



spanwise fluctuations,



streamwise fluctuations,



Optimizing the DIscrete Loss

for inverse problems in fluid mechanics:

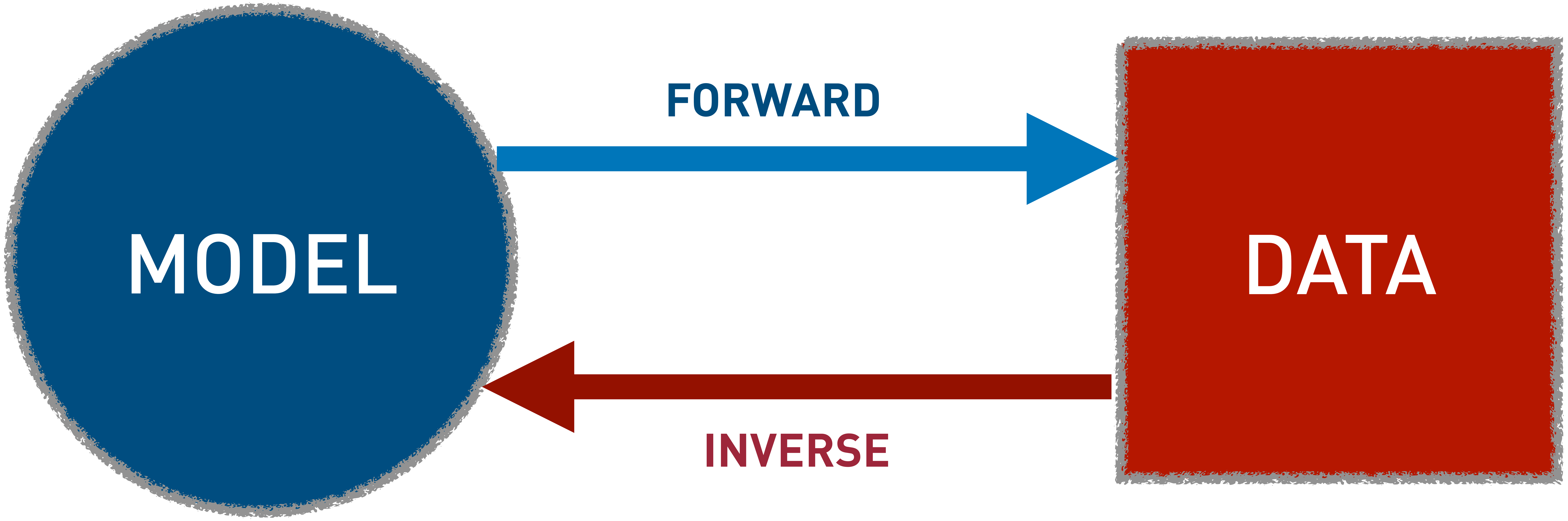
multiresolution and automatic differentiation

Petros Koumoutsakos
Harvard University

WITH

Petr Karnakov

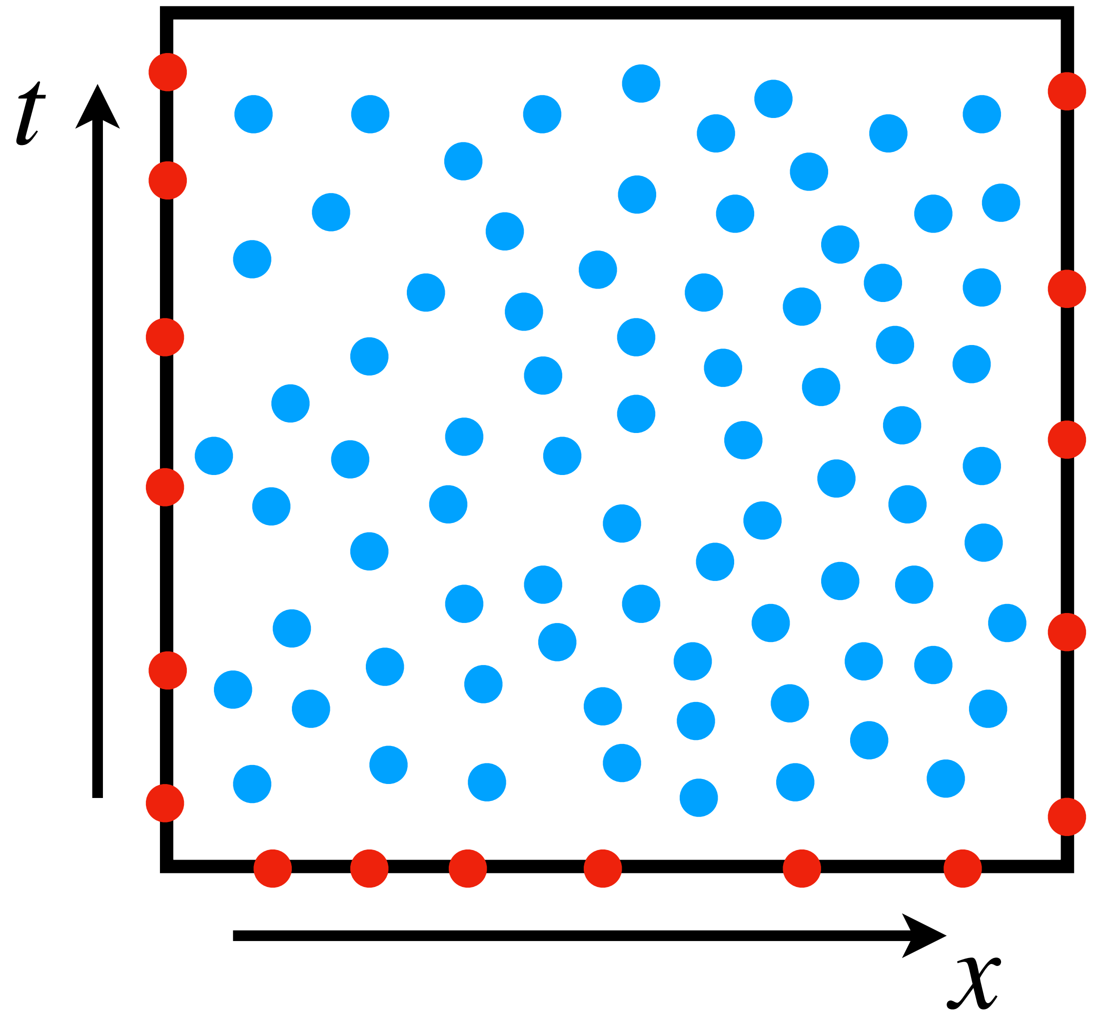
Sergey Litvinov


FORWARD
MODEL
DATA
INVERSE

NNs for PDEs - Forward and Inverse Problems

Equation and boundary conditions

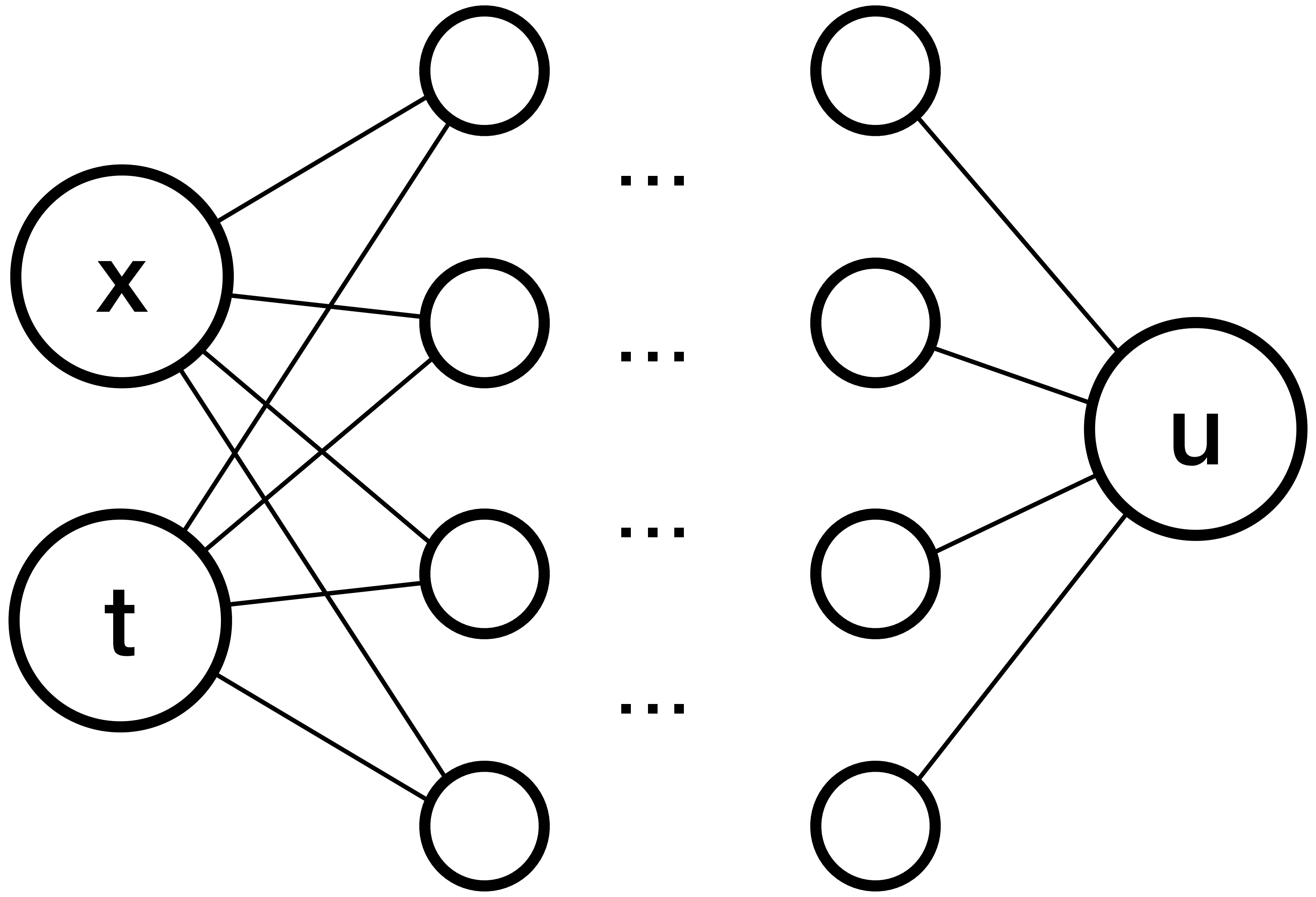
$$\begin{aligned} G[u] &= 0, \quad (x,t) \in \Omega \\ u &= g, \quad (x,t) \in \partial\Omega \end{aligned}$$



Solution as NN: $u(x,t,\theta)$

Loss function

$$L(\theta) = \underbrace{\sum_{(x,t)\in\Omega_1} \Big(G[u](x,t)\Big)^2}_{\text{equation at collocation points } \Omega_1 \subset \Omega} + \underbrace{\sum_{(x,t)\in\Omega_2} \Big(u(x,t,\theta) - g(x,t)\Big)^2}_{\text{boundary conditions at boundary points } \Omega_2 \subset \partial\Omega} \to \min_{\theta}$$



weights θ

COMMUNICATIONS IN NUMERICAL METHODS IN ENGINEERING, Vol. 10, 195–201 (1994)

NEURAL-NETWORK-BASED APPROXIMATIONS FOR SOLVING PARTIAL DIFFERENTIAL EQUATIONS

M. W. M. G. DISSANAYAKE AND N. PHAN-THIEN
Department of Mechanical Engineering, The University of Sydney, Sydney, N.S.W. 2006, Australia

Australia
1994

VOLUME 75, NUMBER 20 PHYSICAL REVIEW LETTERS 13 NOVEMBER 1995

Neural Network Differential Equation and Plasma Equilibrium Solver

B. Ph. van Milligen, V. Tribaldos, and J. A. Jiménez
Asociación EURATOM-CIEMAT para Fusión, Avenida Complutense 22, 28040 Madrid, Spain
(Received 10 March 1995)

Spain
1995

IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 9, NO. 5, SEPTEMBER 1998 987

Artificial Neural Networks for Solving Ordinary and Partial Differential Equations

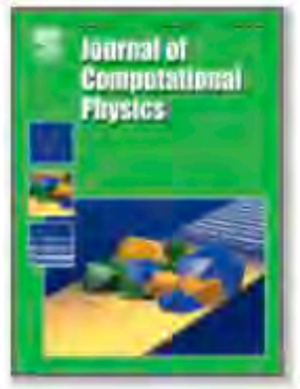
Isaac Elias Lagaris, Aristidis Likas, *Member, IEEE,* and Dimitrios I. Fotiadis

Greece
1998

ELSEVIER

Journal of Computational Physics

Volume 378, 1 February 2019, Pages 686-707



Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations

M. Raissi [a], P. Perdikaris [b], G.E. Karniadakis [a]

PINNs
(2019)

“Although **similar ideas** for constraining neural networks using physical laws have been explored in previous studies [15], [16], here we revisit them using **modern computational tools**, and apply them to **more challenging dynamic problems** described by time-dependent nonlinear partial differential equations”

PINNs WORK: Kuramoto-Sivashinsky 1D ($T_{\text{final}} = 5$, $L = 44$)

increasing network size and varying seed

PINN solution converges to exact

PINNS DO NOT work: Kuramoto-Sivashinsky 1D ($T_{\text{final}} = 100$, $L = 18$)

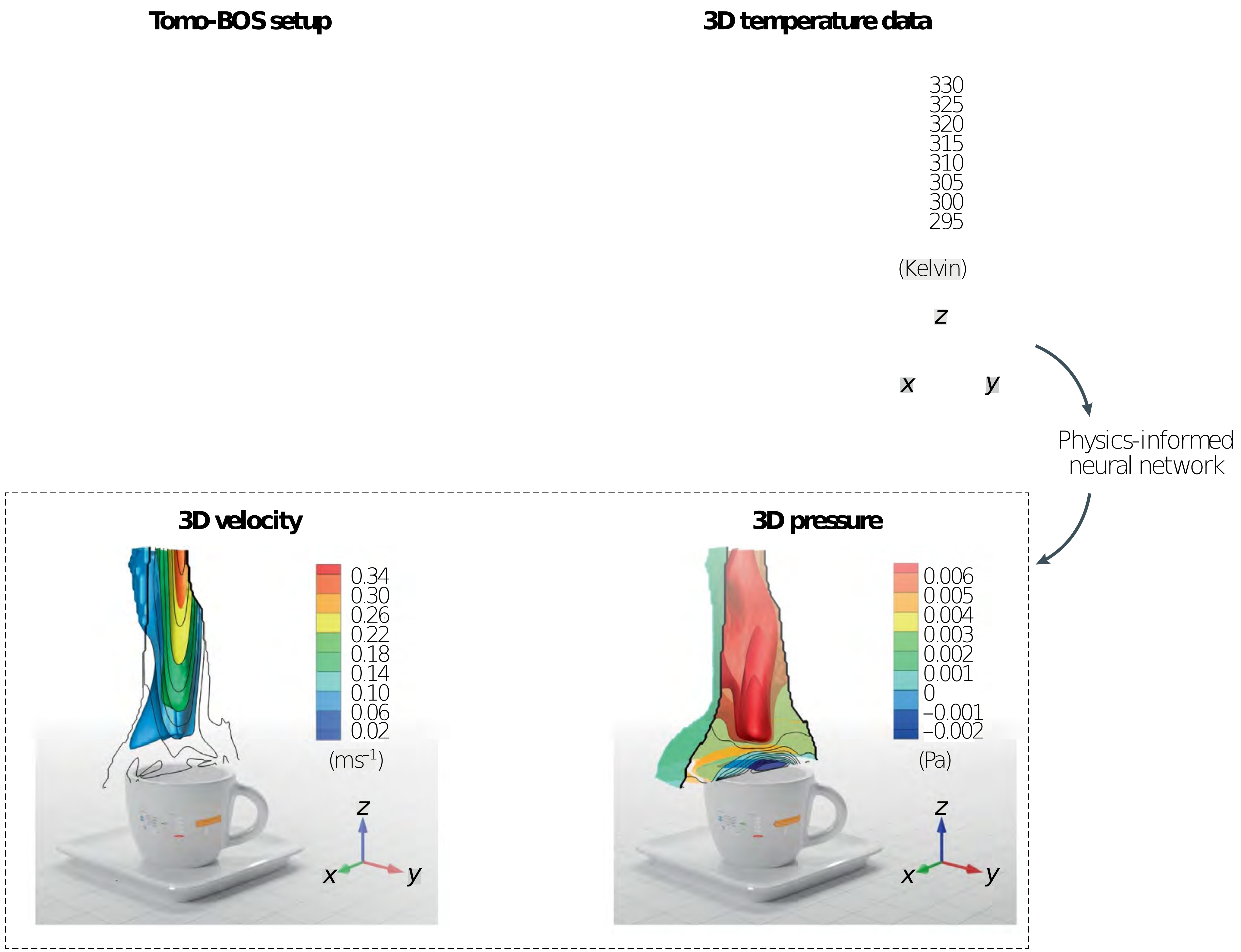
increasing network size and varying seed

PINN fails to converge
for more stiff problem and longer times

PINNS for inverse and ill-posed problems

Loss function may include equations for all fields, and omit unknown data

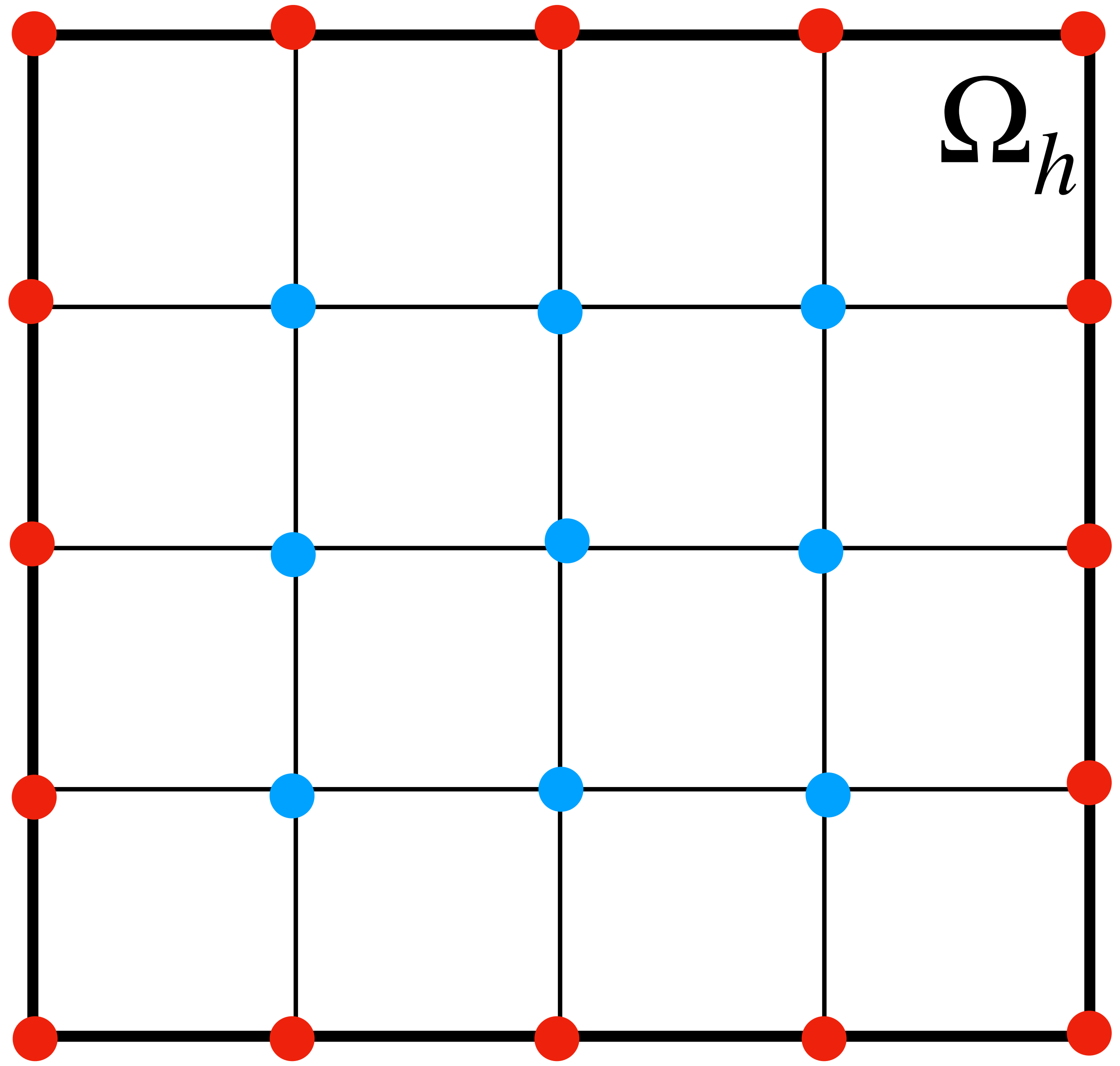
$$
\begin{aligned}
L(\theta) = &\sum_{(x,t)\in\Omega_1} \left(G_u[u](x,t,\theta)\right)^2 \\
&+ \sum_{(x,t)\in\Omega_1} \left(G_v[v](x,t,\theta)\right)^2 \\
&+ \sum_{(x,t)\in\Omega_2} \left(u(x,t,\theta) - u_{\mathrm{ref}}(x,t)\right)^2
\end{aligned}
$$



Cai S, Wang Z, Fuest F, Jeon YJ, Gray C, Karniadakis GE.
Flow over an espresso cup: inferring 3-D velocity and pressure fields from tomographic background oriented Schlieren via physics-informed neural networks.
Journal of Fluid Mechanics. 2021

Optimizing DIscrete Loss - ODIL

- Continuous $F(u) = 0 \qquad u = g\Big|_{\partial\Omega}$
- Discretize $F^h(u^h) = 0 \quad u^h = g^h\Big|_{\partial\Omega}$
  - (often a linear operation $Au^h - b = 0$)



- **Formulate as a discrete loss**

$$L(u^h) = \|F^h(u^h)\|^2 + \|u^h - g^h\|^2$$

- **Optimize the DIscrete Loss**

OPTIMIZATION $\min_{u^h} \; L(u^h)$

INVERSE PROBLEMS: Loss function may include equations and reference data (or omit unknown data)

Solving inverse problems for PDEs

A PARTIAL LIST OF Related works

- differentiable solvers, adjoints : [Amos B, Kolter JZ. Optnet: Differentiable optimization as a layer in neural networks. ICML, 2017]
- discretize-then-optimize: [Gunzburger MD. Perspectives in flow control and optimization. SIAM, 2002]
- direct collocation method: [Von Stryk O. Numerical solution of optimal control problems by direct collocation. 1993]
- variational data assimilation: [Rabier F et al. The ECMWF operational implementation of four-dimensional variational assimilation. 2000]

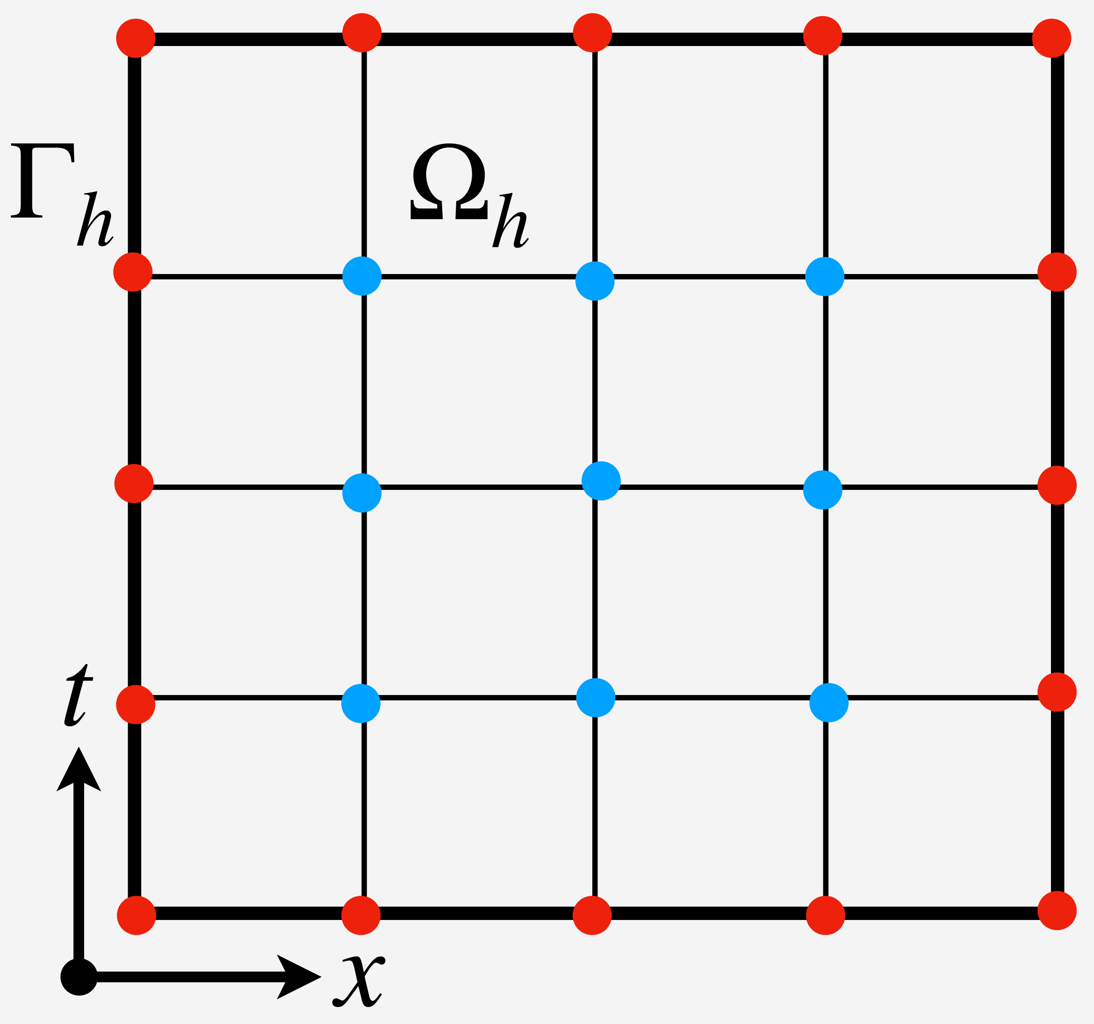
Optimizing a DIscrete Loss

- Variant of penalty method with PDEs discretized on a grid
- Automatic differentiation and standard gradient-based optimizers
- Multigrid technique that accelerates optimization
- **Faster and more accurate than PINN; simpler and more versatile than adjoints**

ODIL vs PINN

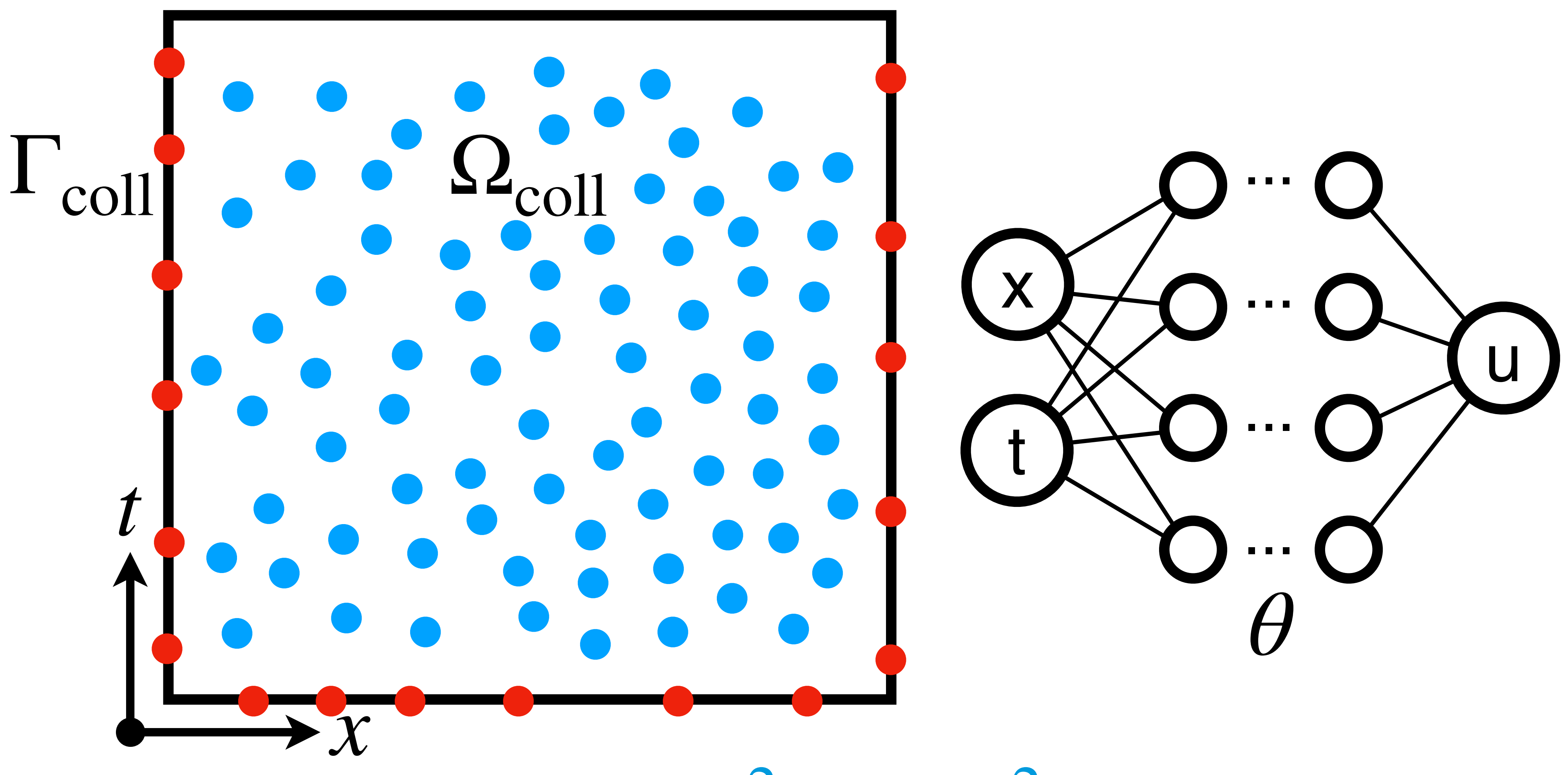
- Wave equation $\left.\dfrac{\partial^2 u}{\partial t^2} - \dfrac{\partial^2 u}{\partial x^2} = 0\right|_\Omega$ with conditions $\left.u = g\right|_\Gamma$

- **ODIL solution is discrete field u_i^n**



$$L(u) = \sum_{(i,n)\in\Omega_h} \left(\frac{u_i^{n-1} - 2u_i^n + u_i^{n+1}}{\Delta t^2} - \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{\Delta x^2} \right)^2 + \sum_{(i,n)\in\Gamma_h} \left(u_i^n - g_i^n\right)^2 \to \min_u$$

- **PINN: solution is neural network $u_\theta(x,t)$**

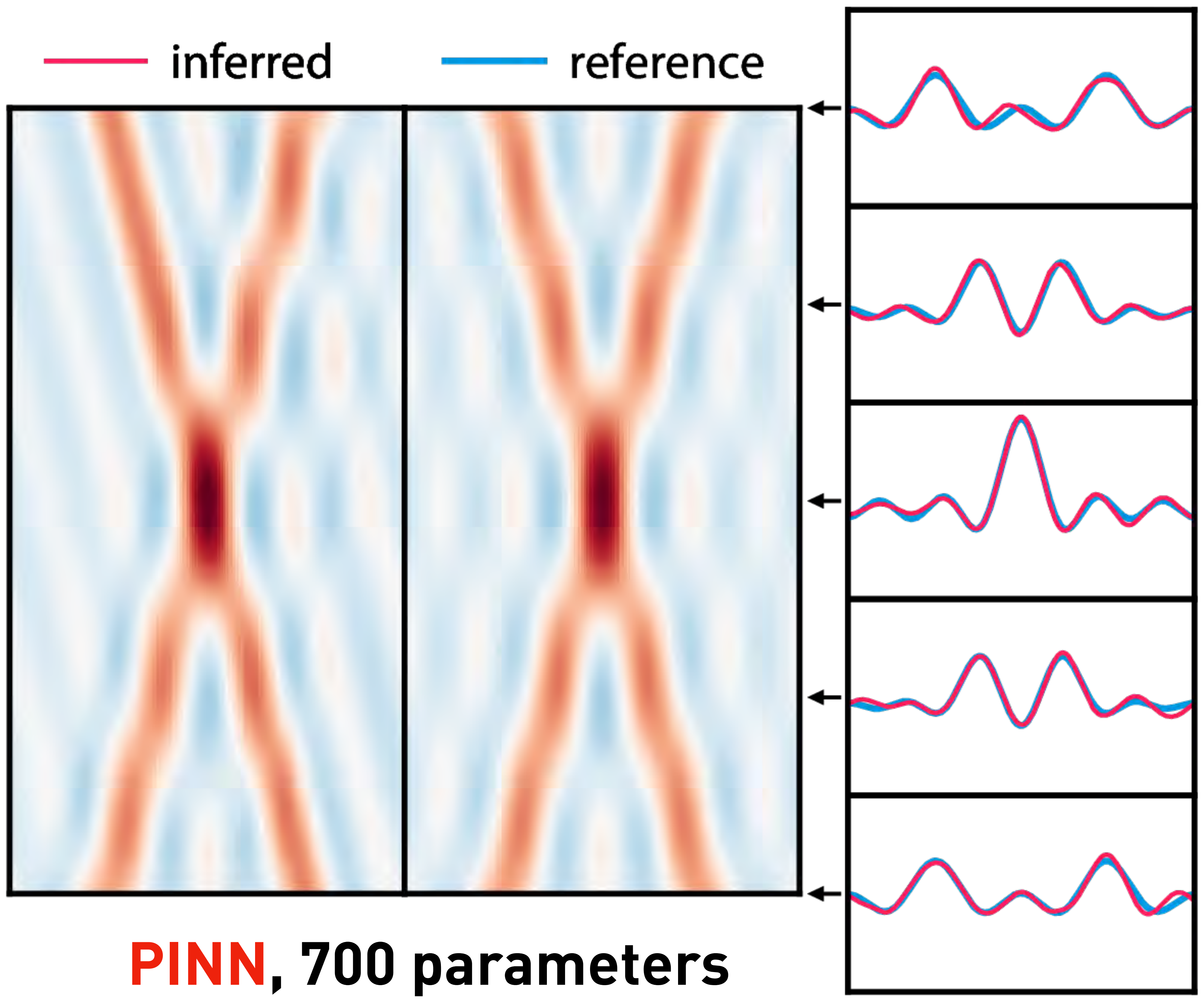


$$L(\theta) = \sum_{(x,t)\in\Omega_{\text{coll}}} \left(\frac{\partial^2 u_\theta}{\partial t^2} - \frac{\partial^2 u_\theta}{\partial x^2} \right)^2 + \sum_{(x,t)\in\Gamma_{\text{coll}}} \left(u_\theta(x,t) - g(x,t)\right)^2 \to \min_\theta$$

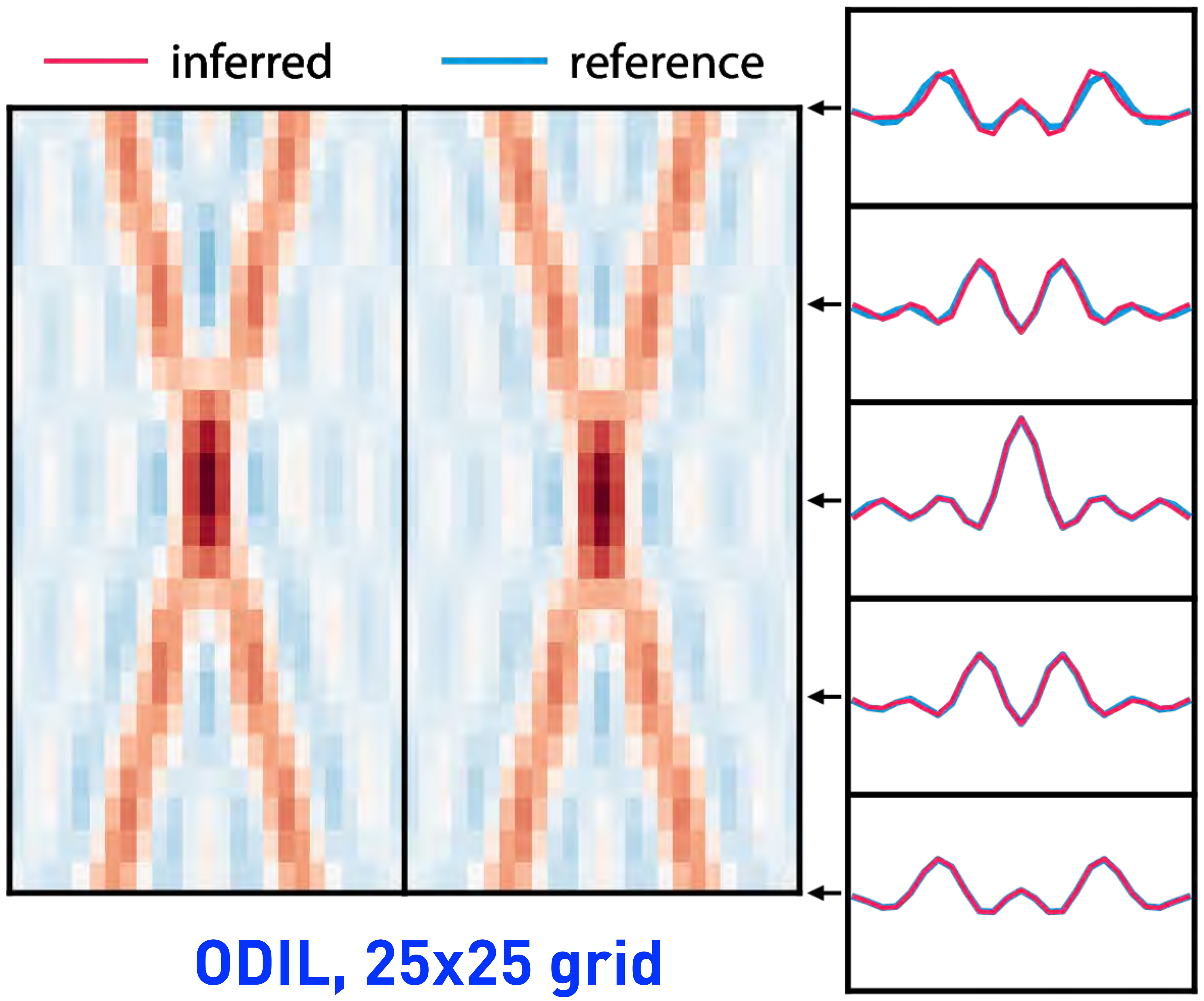
PINN vs ODIL

Wave equation $\frac{\partial^2 u}{\partial t^2} - \frac{\partial^2 u}{\partial x^2} = 0$

$$u(x,t) = \sum_{k=1}^{5} (\cos(x - t + 0.5)\pi k + \cos(x + t + 0.5)\pi k$$

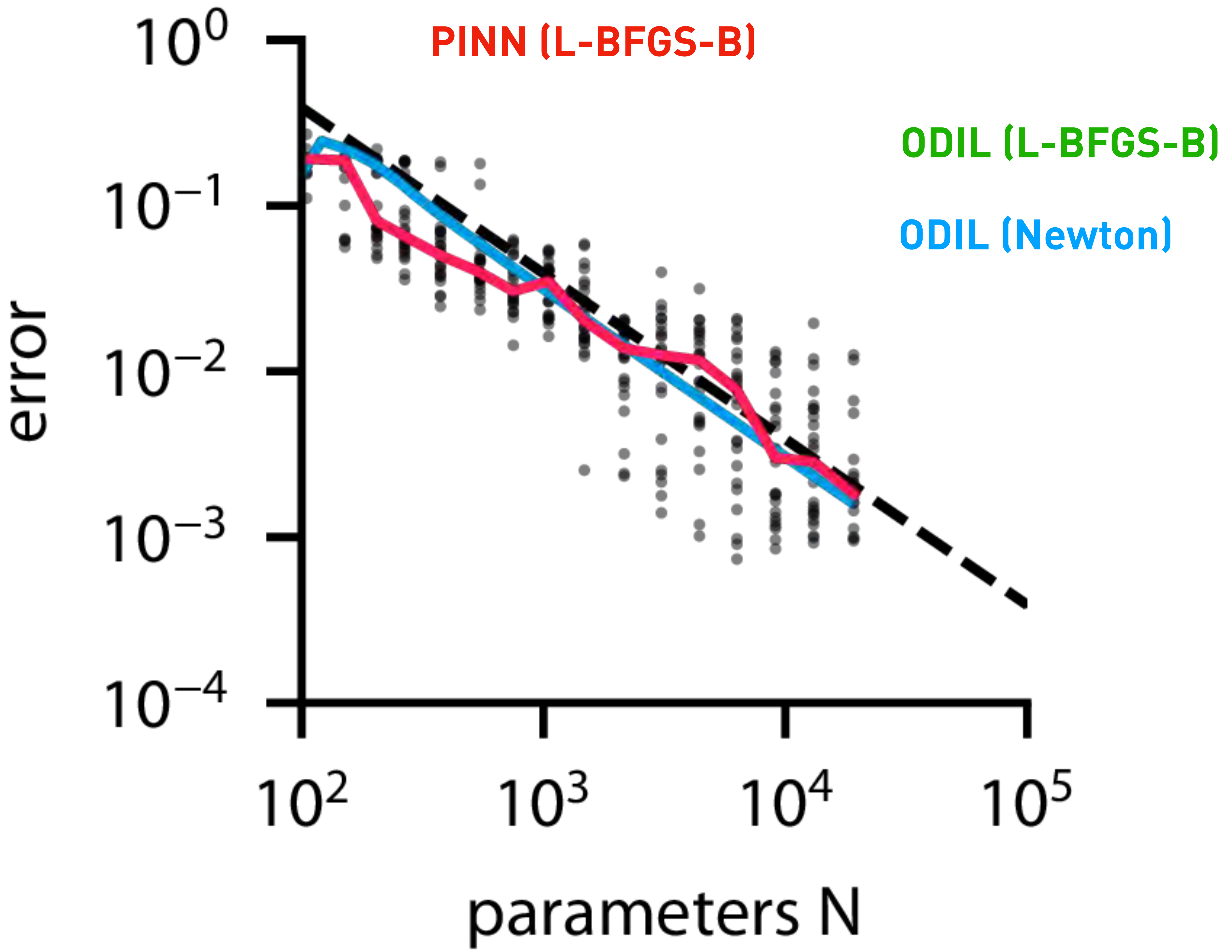


PINN, 700 parameters



ODIL, 25x25 grid

PINN vs ODIL (1D Wave Equation)

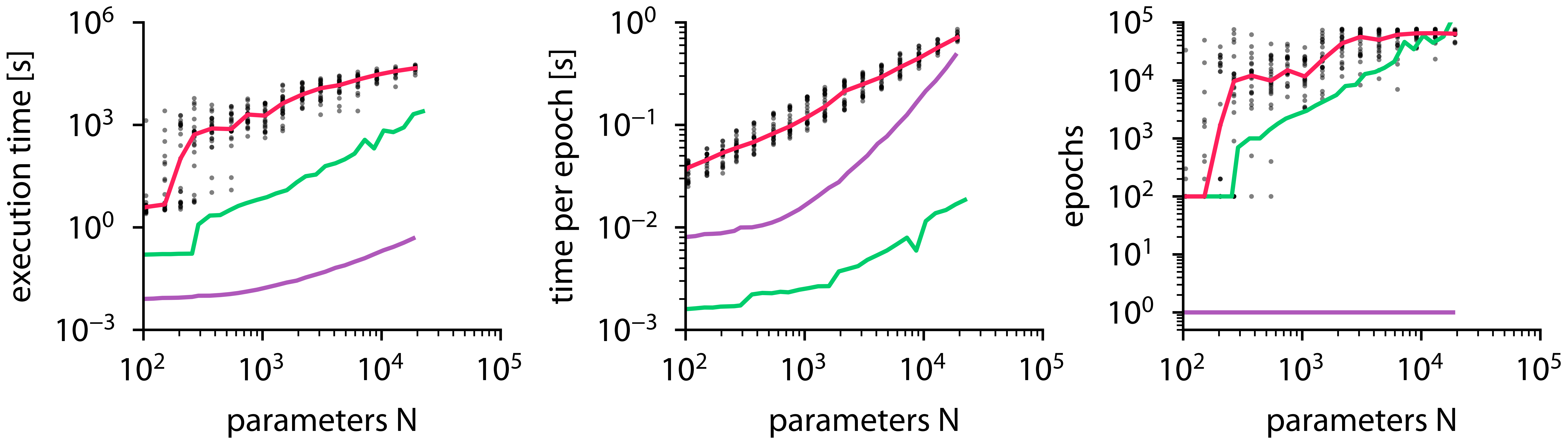


PINN accuracy depends on initialisation, etc.

ODIL x100'000 FASTER than **PINNs**

Cost of PINN and ODIL

Total cost = **Cost per epoch** × **Number of epochs**



	Total cost		Cost per epoch		Number of epochs
PINN (L-BFGS)	10’000 s	=	0.2 s	×	50’000
ODIL (L-BFGS)	50 s	=	0.005 s	×	10’000
ODIL (Newton)	0.05 s	=	0.05 s	×	1

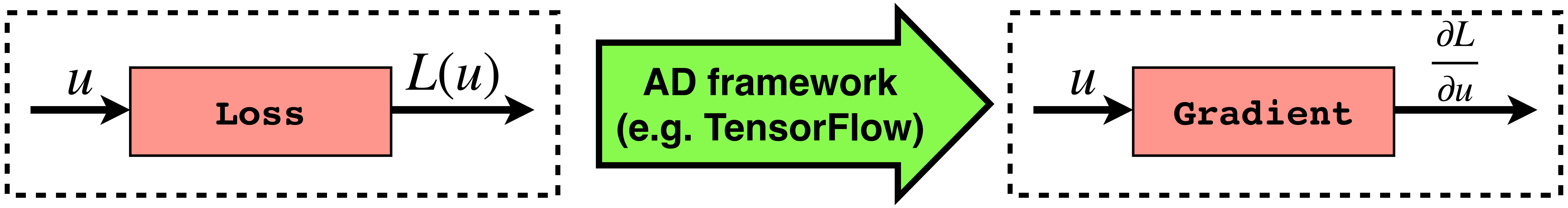
measurements for $N \approx 3000$ on one CPU core

ODIL (L-BFGS) **x100 faster** than PINN

ODIL (Newton) **x100’000 faster** than PINN

Automatic differentiation

- ODIL



- mODIL

mODIL and **ODIL** use the same implementation of loss function

mODIL: Multiresolution ODIL

- ODIL loss function on a grid of $N \times N$ points

$$L(u) = \sum_{(i,n)\in\Omega_h} \left(\frac{u_i^{n-1} - 2u_i^n + u_i^{n+1}}{\Delta t^2} - \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{\Delta x^2} \right)^2 + \sum_{(i,n)\in\Gamma_h} (u_i^n - g_i^n)^2$$

- Use a hierarchy of M levels, e.g. $N = N_1 = 17,\ N_2 = 9,\ N_3 = 5\ ,\ \ N_M = 3$
- **Multigrid decomposition** of the unknown field

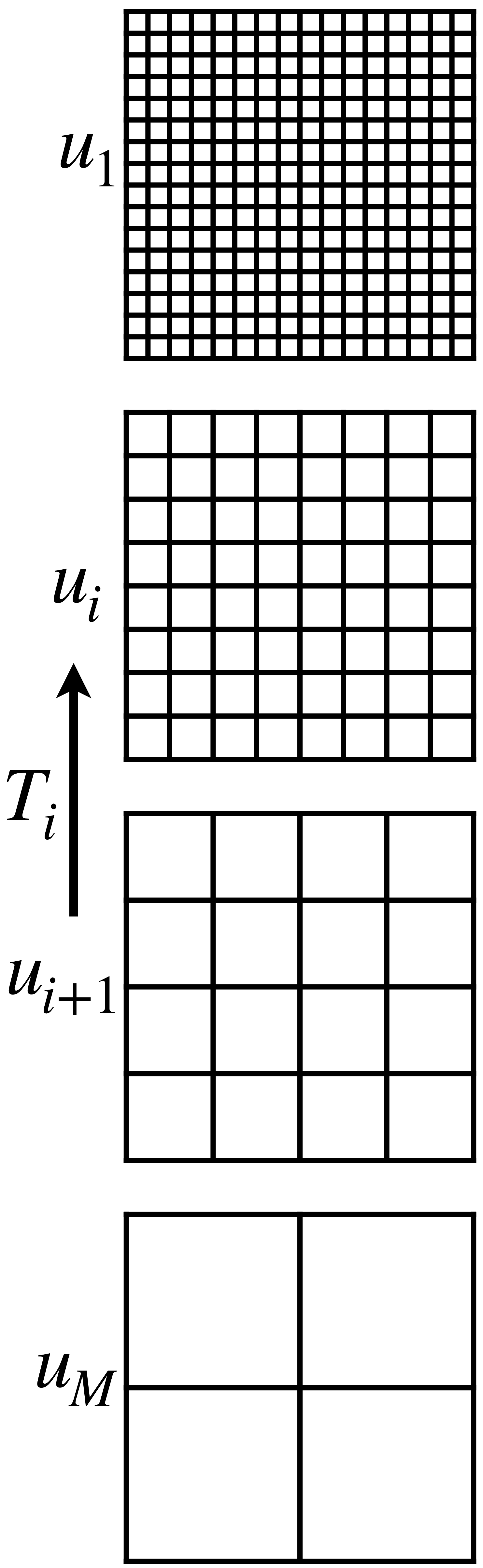
$$u = u_1 + T_1 u_2 + T_1 T_2 u_3 + \ldots + T_1 \ldots T_{M-1} u_M$$

  with interpolation operators T_i

- Instead of $L(u)$, **mODIL** minimizes

$$L_M(u_1, \ldots, u_M) := L(u_1 + T_1 u_2 + T_1 T_2 u_3 + \ldots + T_1 \ldots T_{M-1} u_M)$$

- Same as in ODIL: **keep using automatic differentiation**

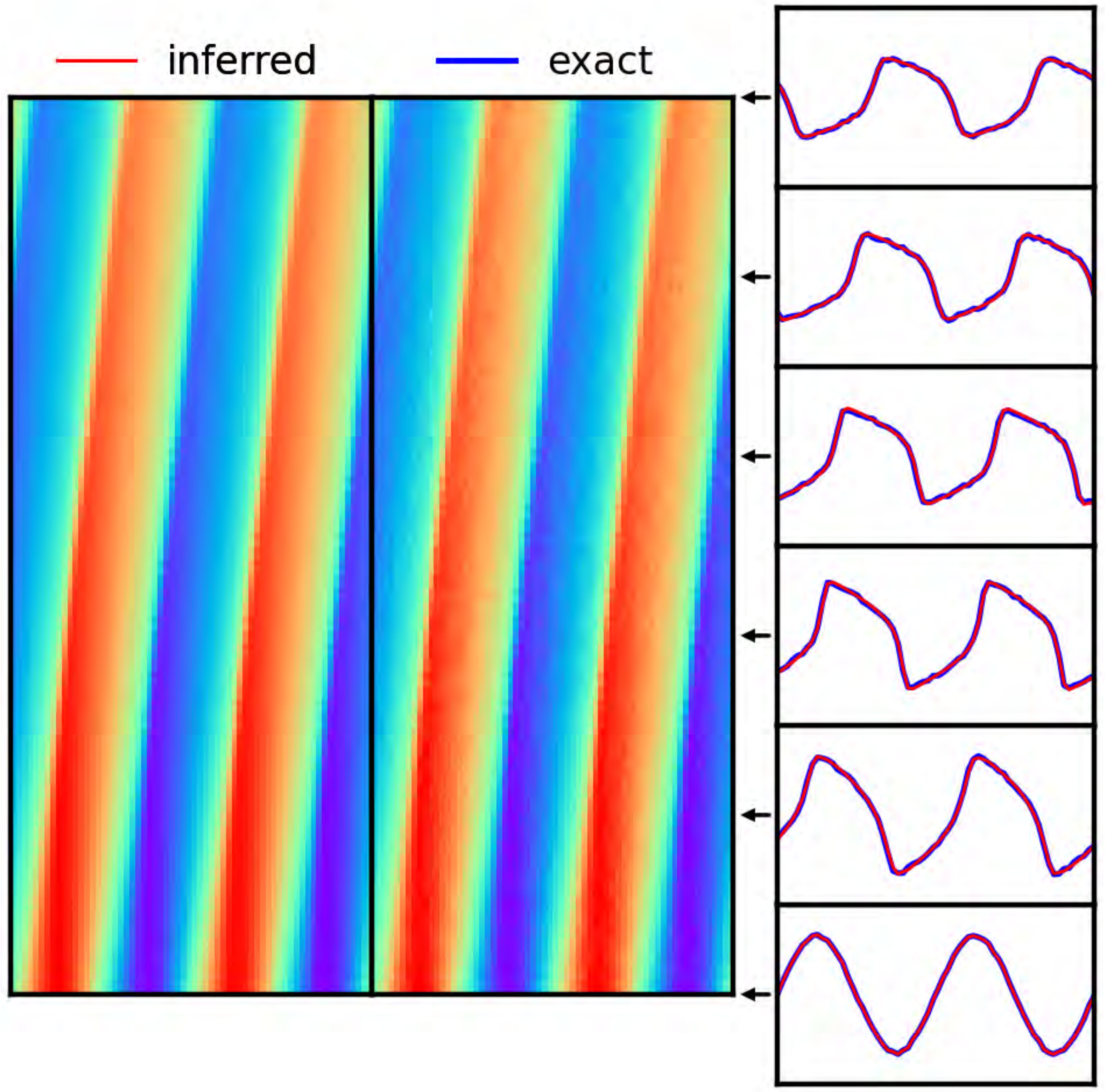


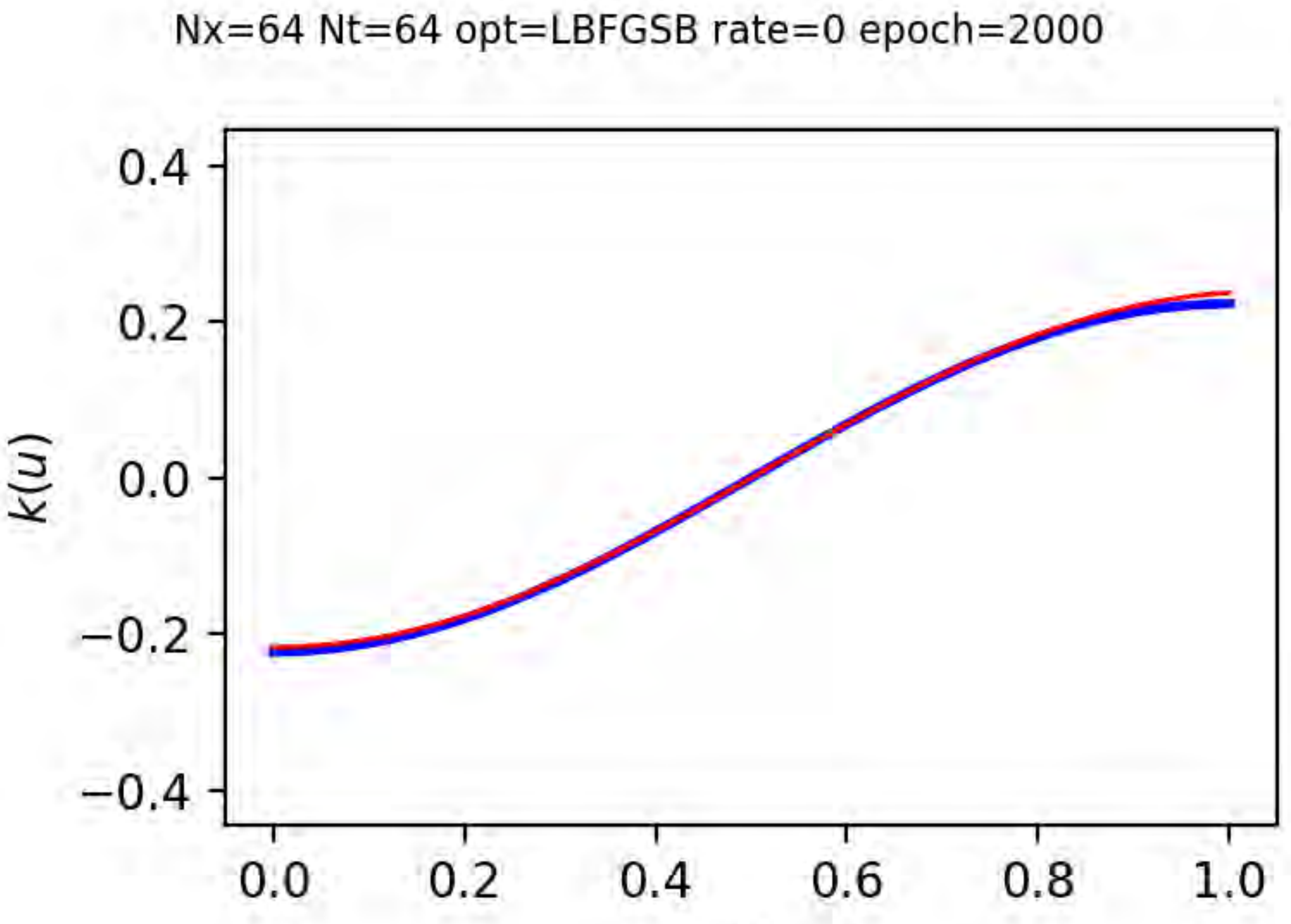
ODIL: Inferring a nonlinear flux

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} k(u) = 0$$

GIVEN: initial & final u

UNKNOWN : k(u)



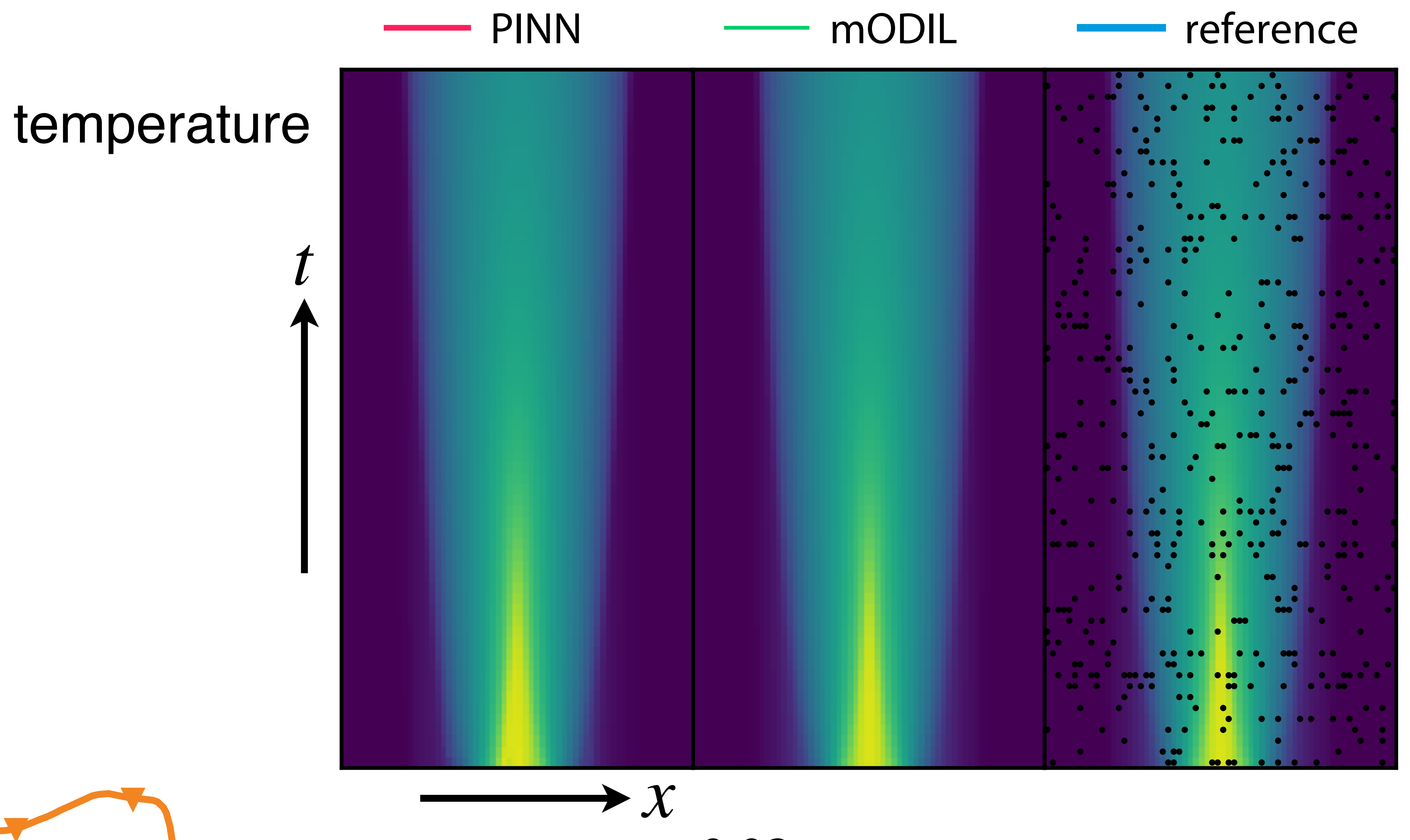


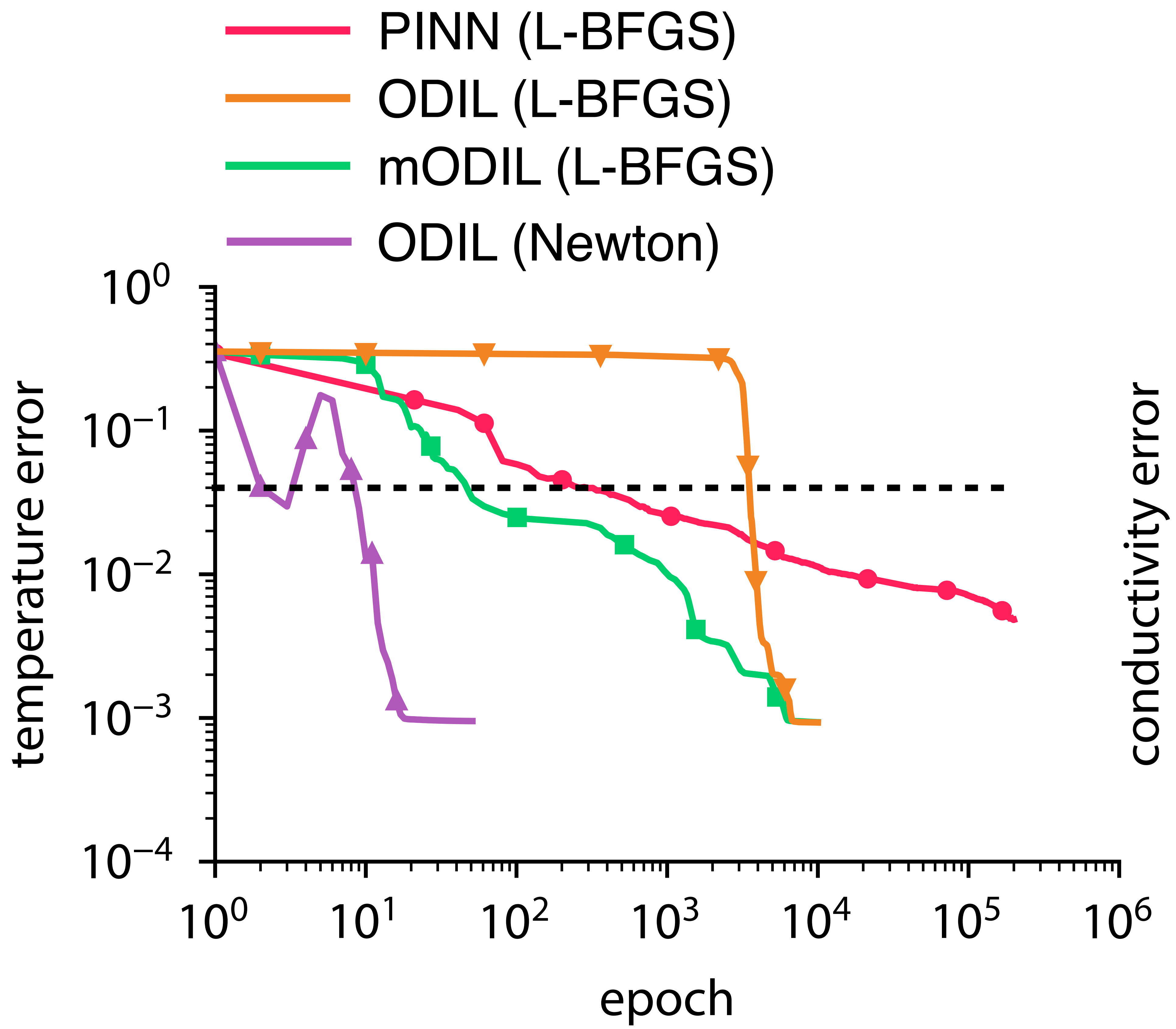
Conductivity from temperature data

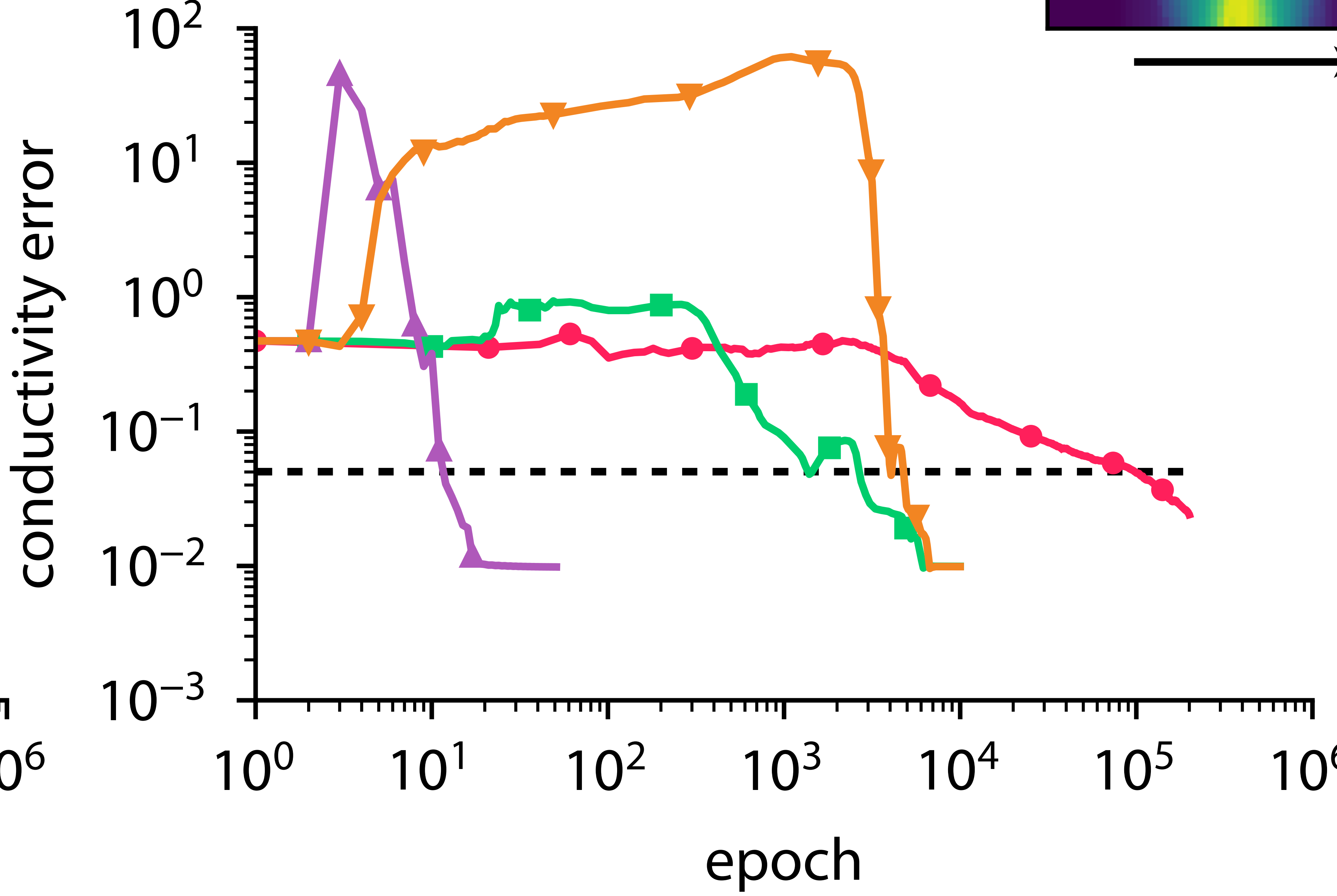
- Infer conductivity $k(u)$ in the heat equation

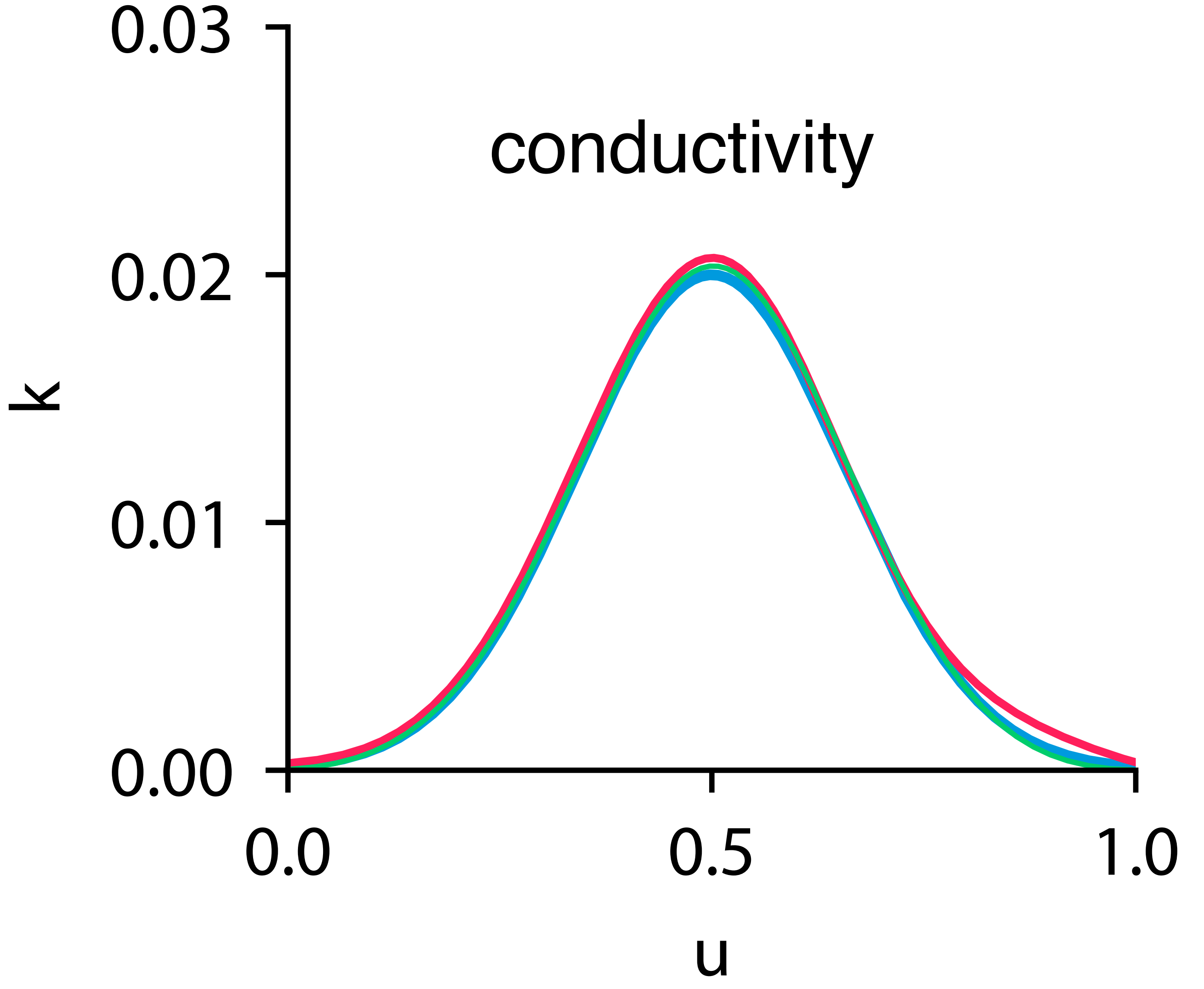
$$\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\left(k(u)\frac{\partial u}{\partial x}\right) = 0$$

$k(u)$ as neural network



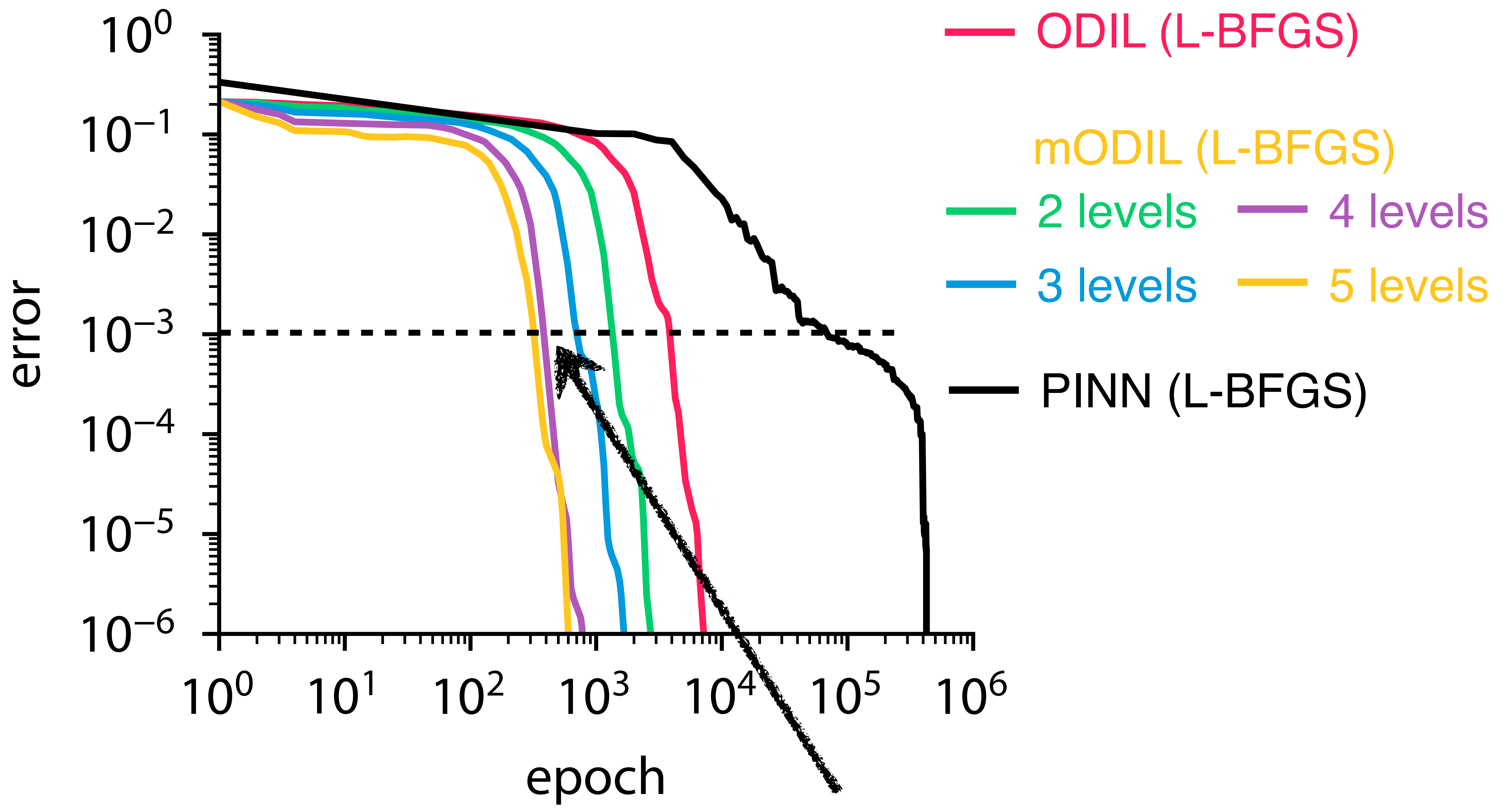


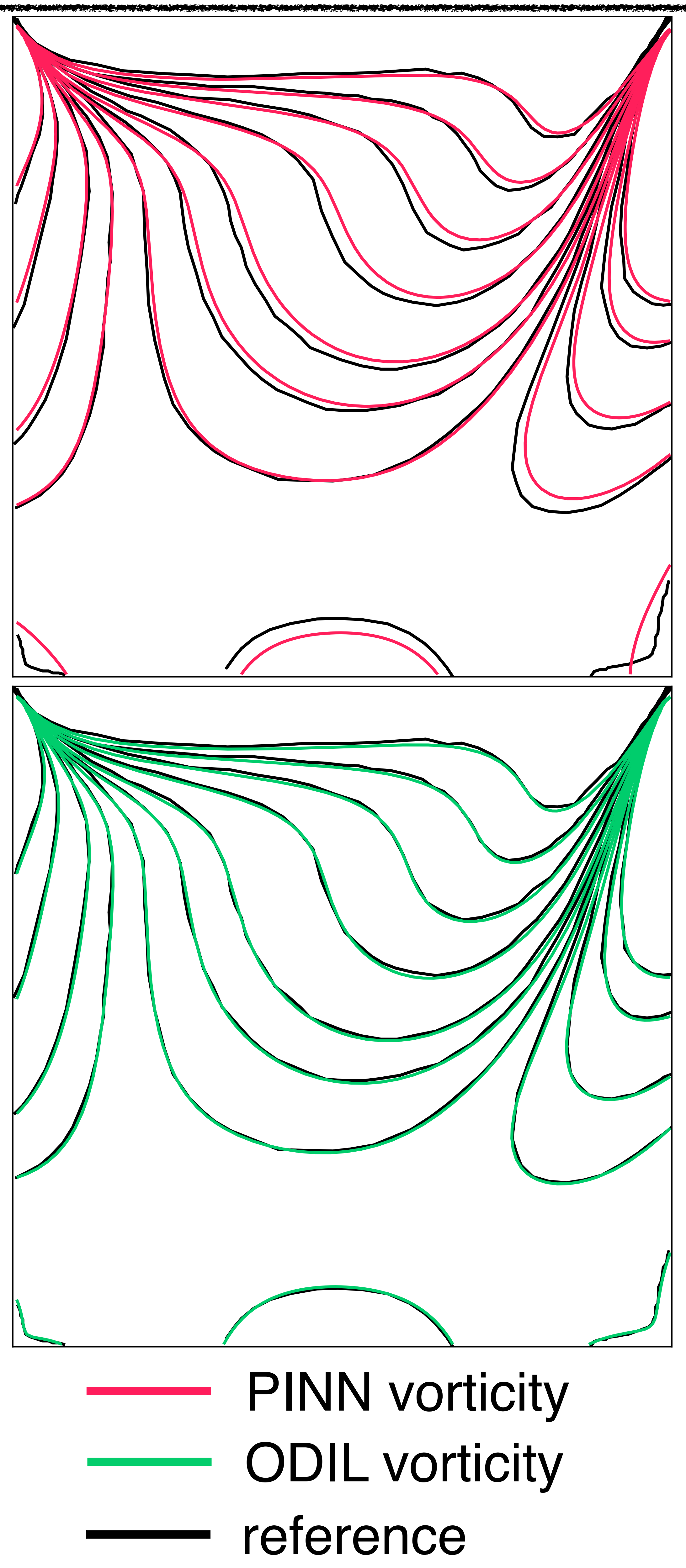




Lid-driven cavity Re=100

More grid levels $\Longrightarrow$ faster convergence

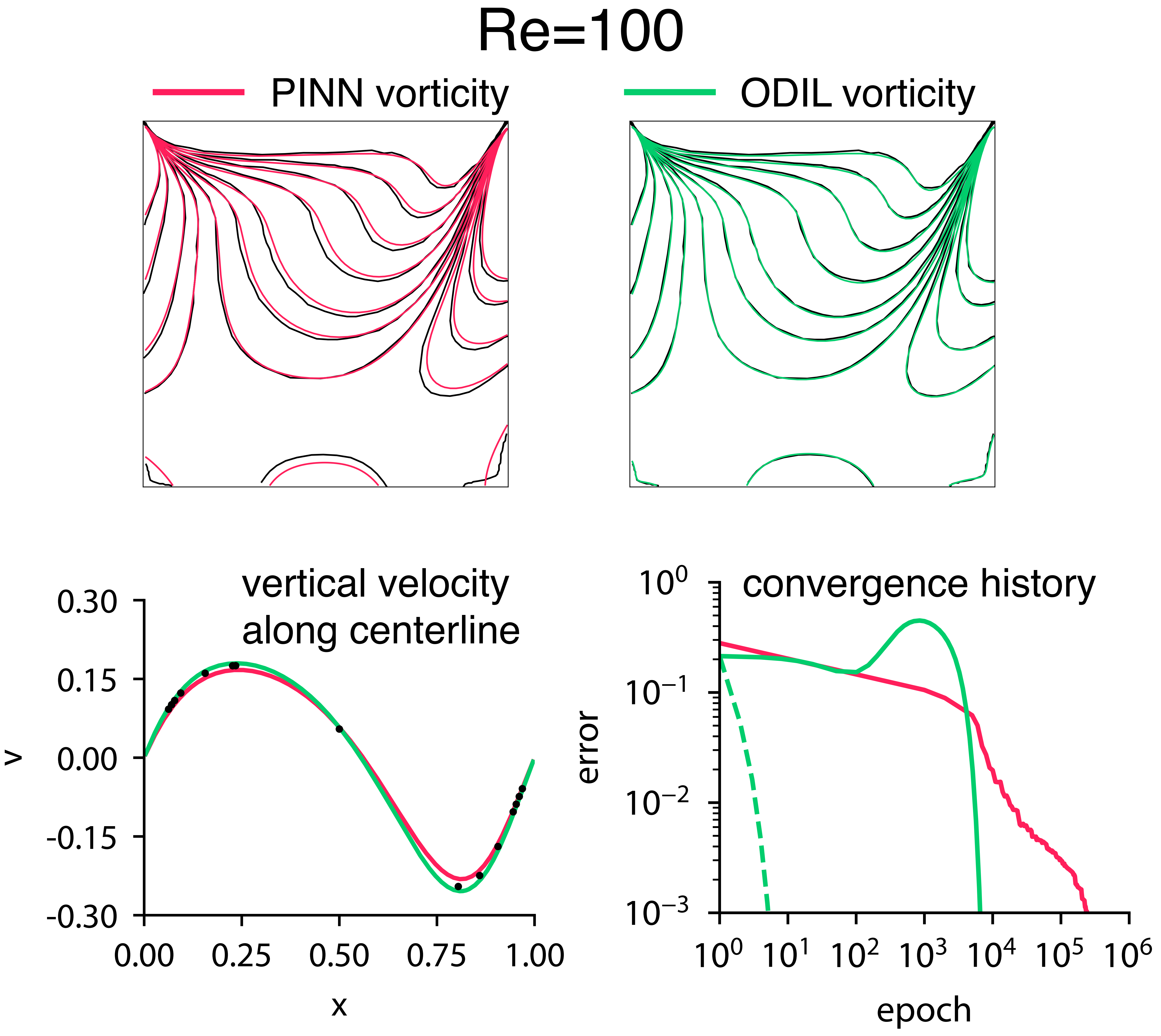


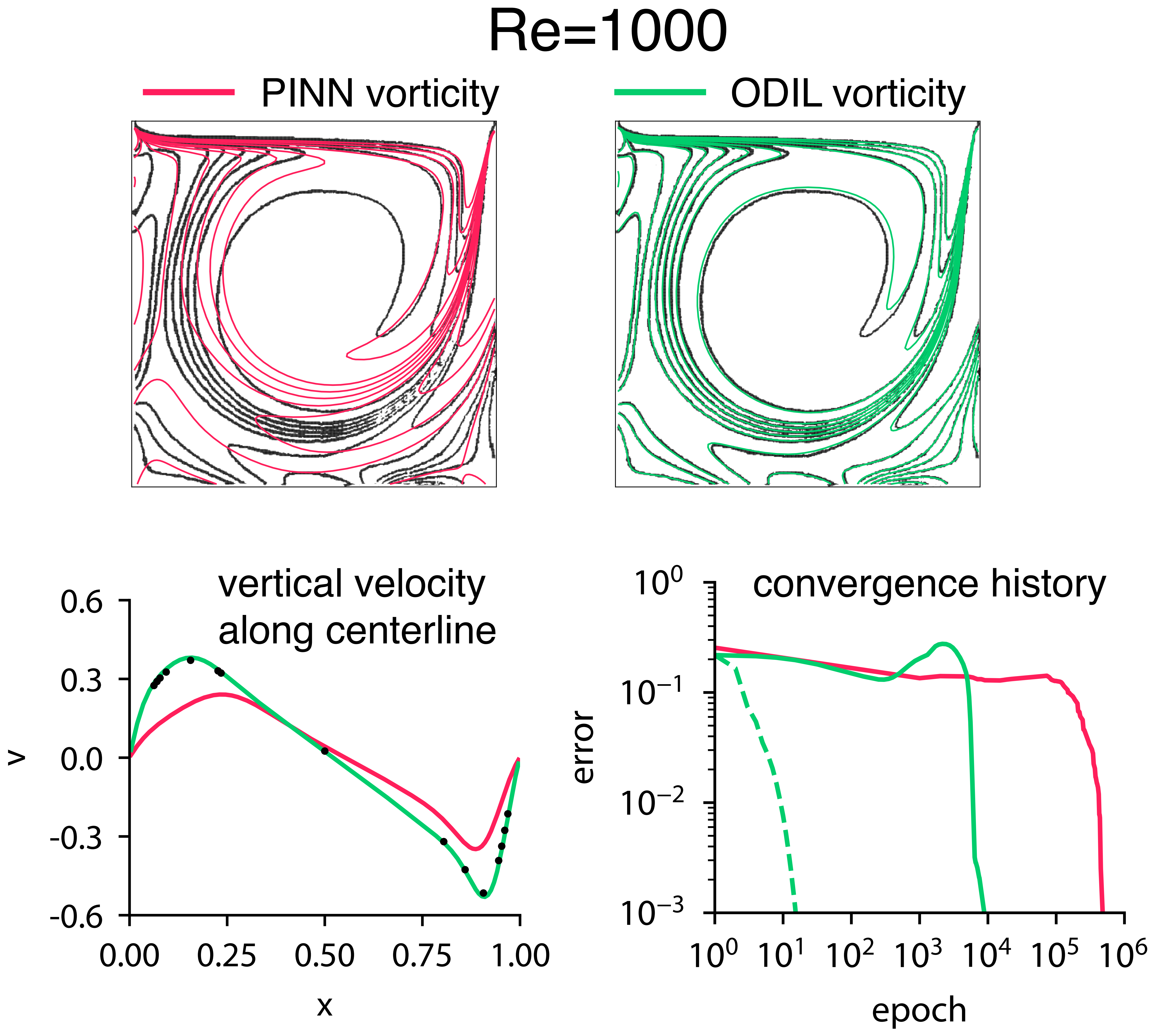


mODIL converges x20 faster than ODIL
mODIL converges x200 faster than PINN

Ghia UK, Ghia KN, Shin CT. JCP. 1982

Lid-driven cavity





PINN fails at higher Re

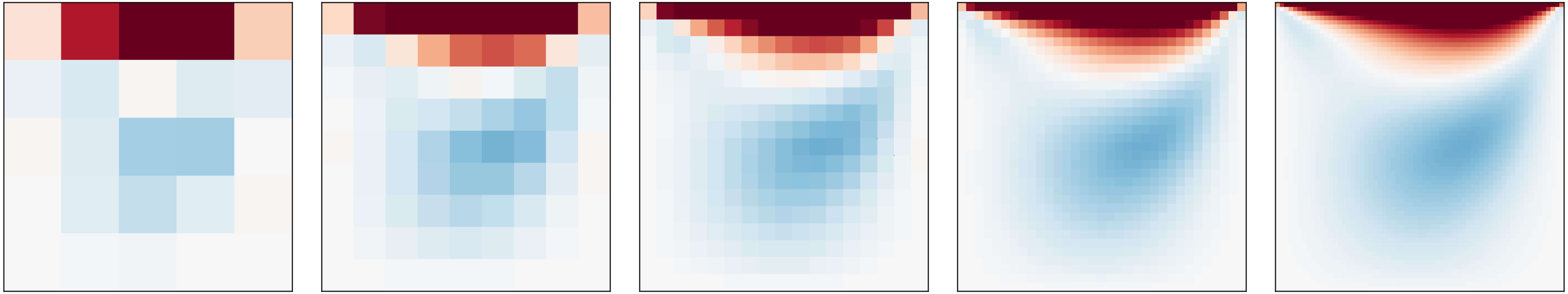
Ghia UK, Ghia KN, Shin CT. JCP. 1982

Lid-driven cavity Re=100

- mODIL: Multigrid decomposition of velocity u using 5 levels

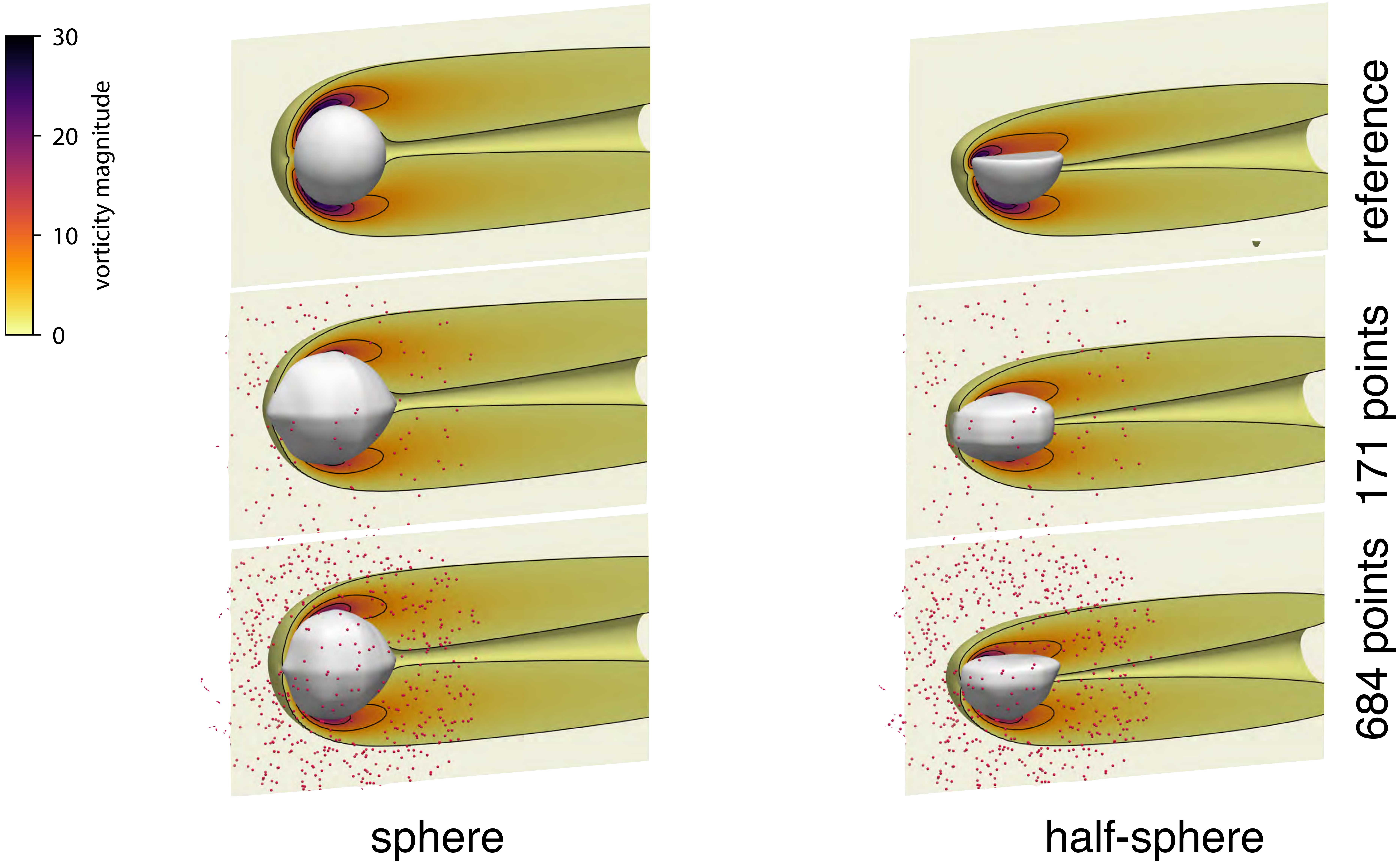
$$u = u_1 + T_1(u_2 + T_2(u_3 + T_3(u_4 + T_4 u_5)))$$

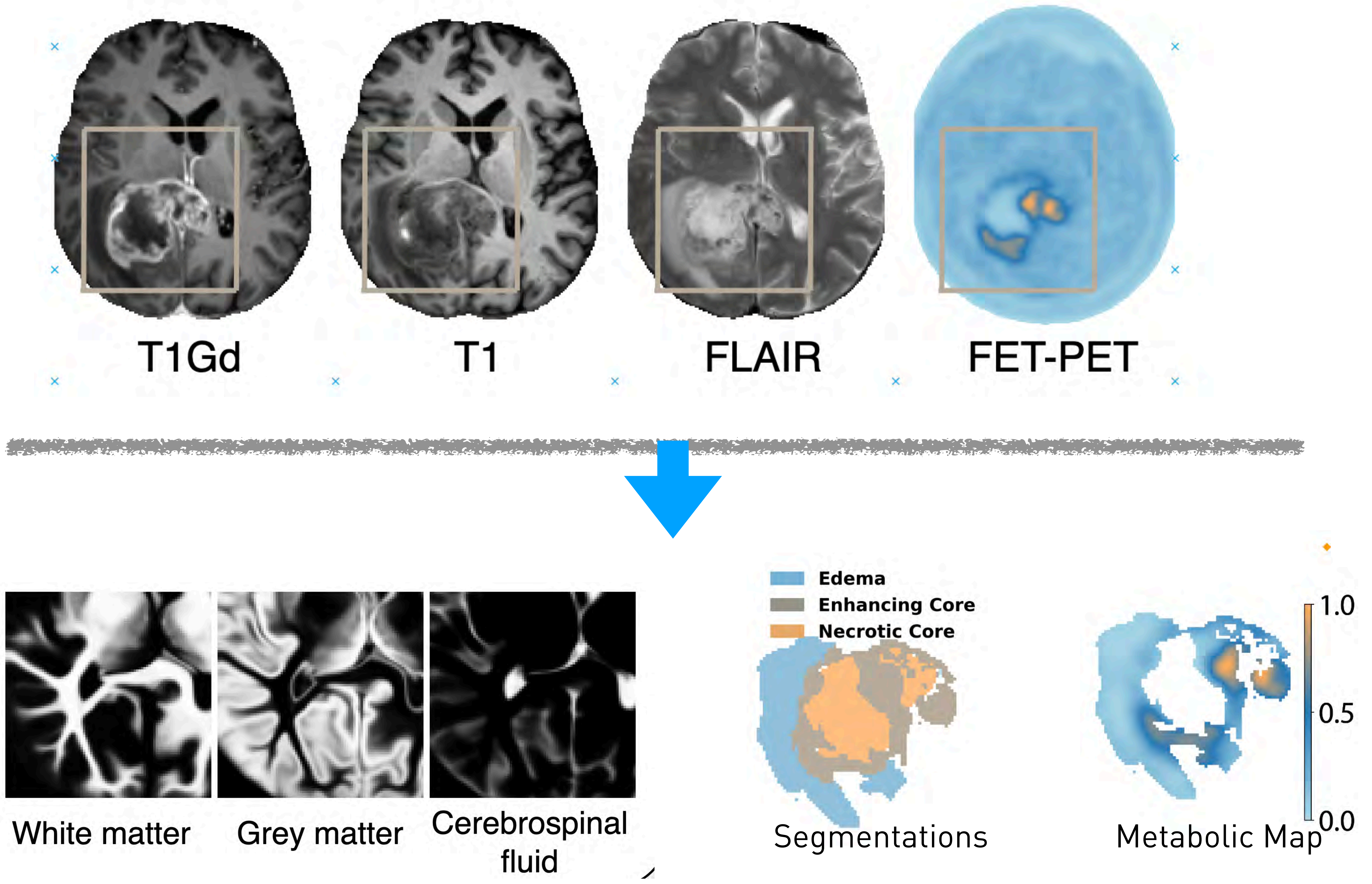
with interpolation operators T_i



u_5 | $u_4 + T_4 u_5$ | $u_3 + T_3(u_4 + T_4 u_5)$ | $u_2 + T_2(u_3 + T_3(u_4 + T_4 u_5))$ | $u_1 + T_1(u_2 + T_2(u_3 + T_3(u_4 + T_4 u_5))) = u$

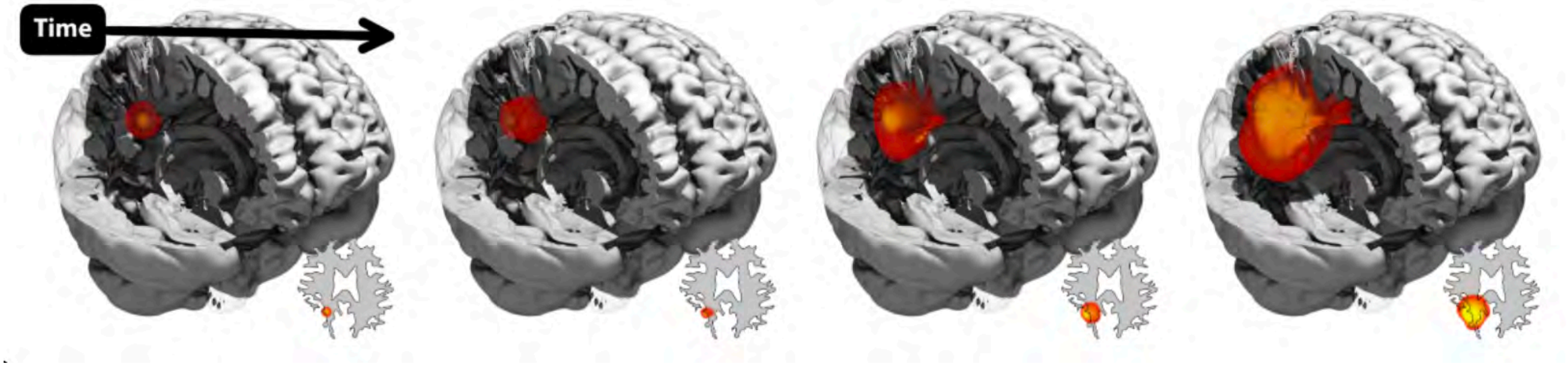
Body shape from velocity




T1Gd
T1
FLAIR
FET-PET
White matter
Grey matter
Cerebrospinal fluid
Edema
Enhancing Core
Necrotic Core
Segmentations
1.0
0.5
0.0
Metabolic Map

$c = c(\vec{x}, t) \qquad D = D_w w(\vec{x}) + D_g g(\vec{x})$

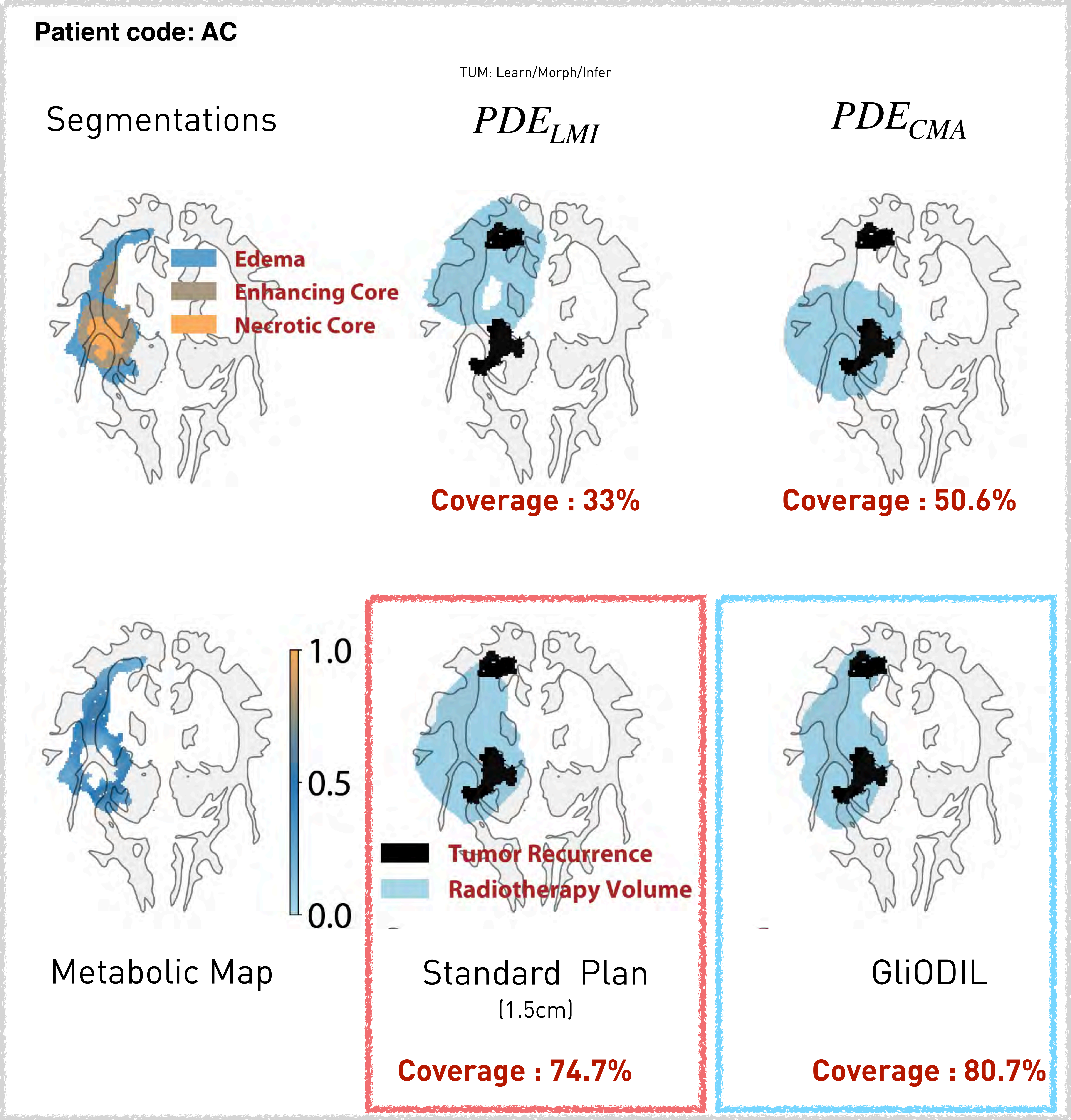
$$\frac{\partial c}{\partial t} = \nabla \cdot (D \nabla c) + \rho c(1 - c)$$ + Brain Mechanics

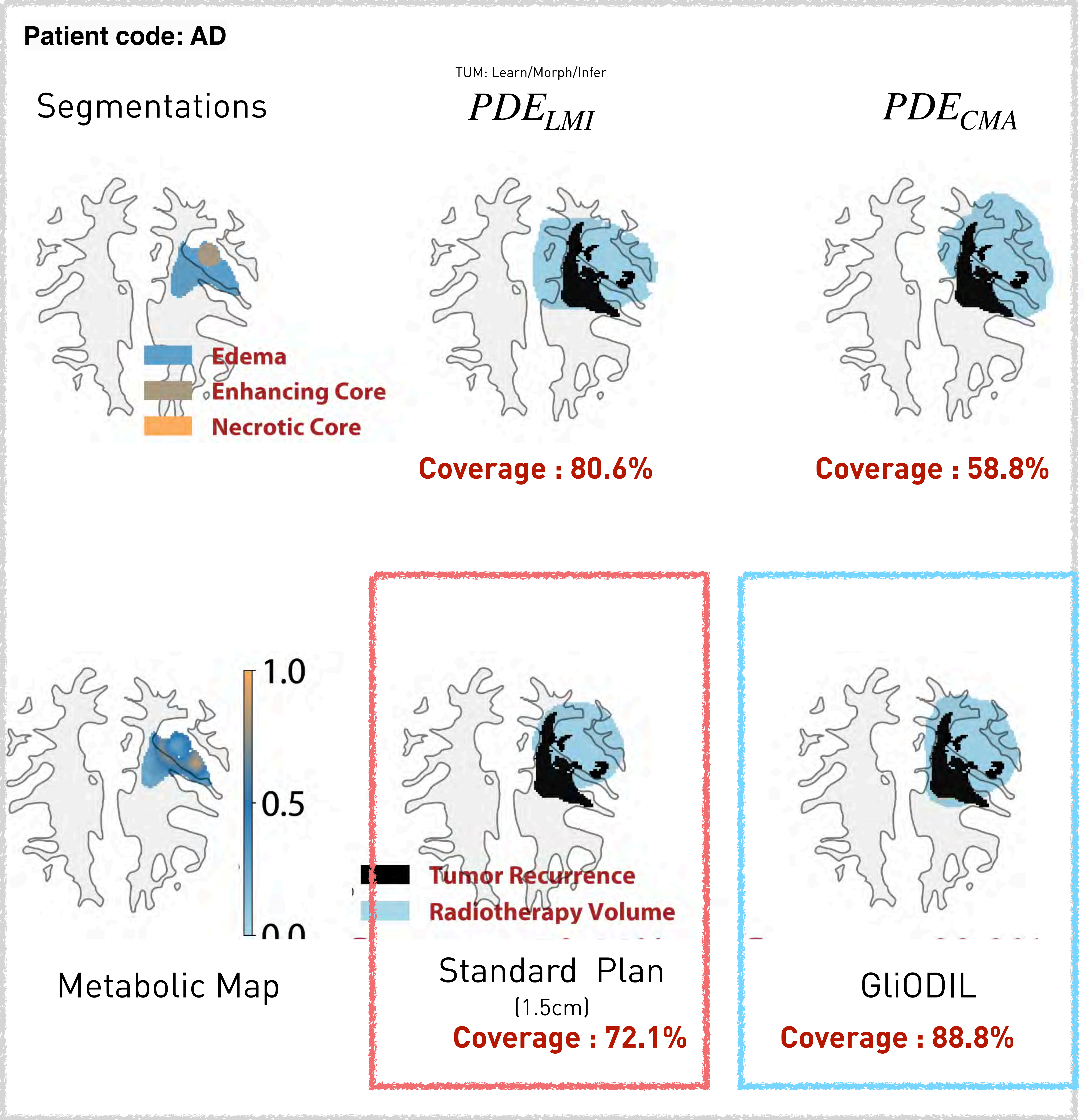


In GliODIL:

1. Discretize Equations
2. Get DATA - Dpt. of Neuroradiology, Klinikum Rechts der Isar, Munich, Germany TranslaTUM, Center for Translational Cancer Research, Munich, Germany
3. Form a Discrete Loss: $\mathscr{L} = L_{physics} + L_{DATA} + L_{ic}$
4. Solve the Inverse problem

Individualizing Glioma Radiotherapy Planning by Data and Physics-Informed Optimization of Discrete Loss: with M. Balcerak, B. Menje (UZH)





SUMMARY FOR ODIL

- ✓ Variant of penalty method with PDEs discretized on a grid
- ✓ Automatic differentiation and standard gradient-based optimizers
- ✓ Multigrid technique that accelerates optimization
- ✓ **ORDERS OF MAGNITUDE FASTER and MORE ACCURATE than PINN;**
- ✓ **Simpler and more versatile than adjoints**
- ➡ **High Dimensions ?**

PNAS nexus
PNAS Nexus, 2024, 3, 1–16
https://doi.org/10.1093/pnasnexus/pgae005
Advance access publication 11 January 2024
Research Report

Solving inverse problems in physics by optimizing a discrete loss: Fast and accurate learning without neural networks

Petr Karnakov[a], Sergey Litvinov[a] and Petros Koumoutsakos[a,*]

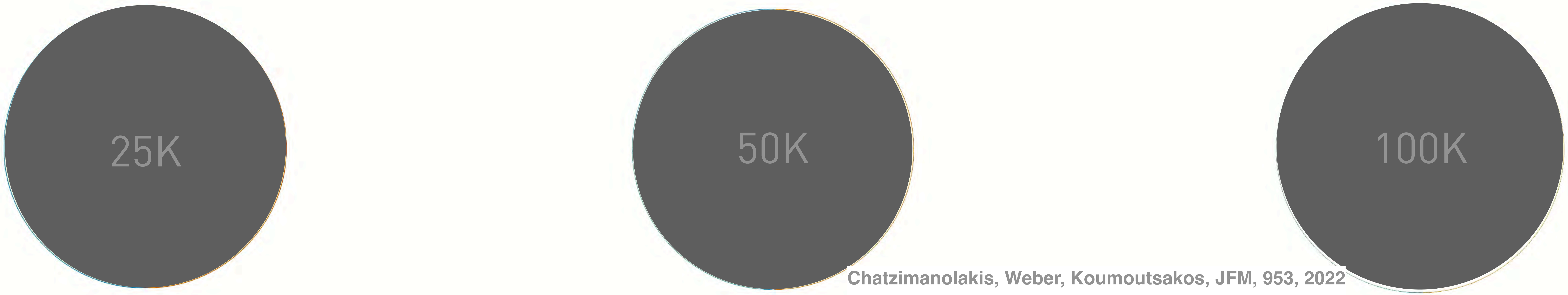
Eur. Phys. J. E (2023) 46:59
https://doi.org/10.1140/epje/s10189-023-00313-7
THE EUROPEAN PHYSICAL JOURNAL E
Regular Article – Flowing Matter

Flow reconstruction by multiresolution optimization of a discrete loss with automatic differentiation

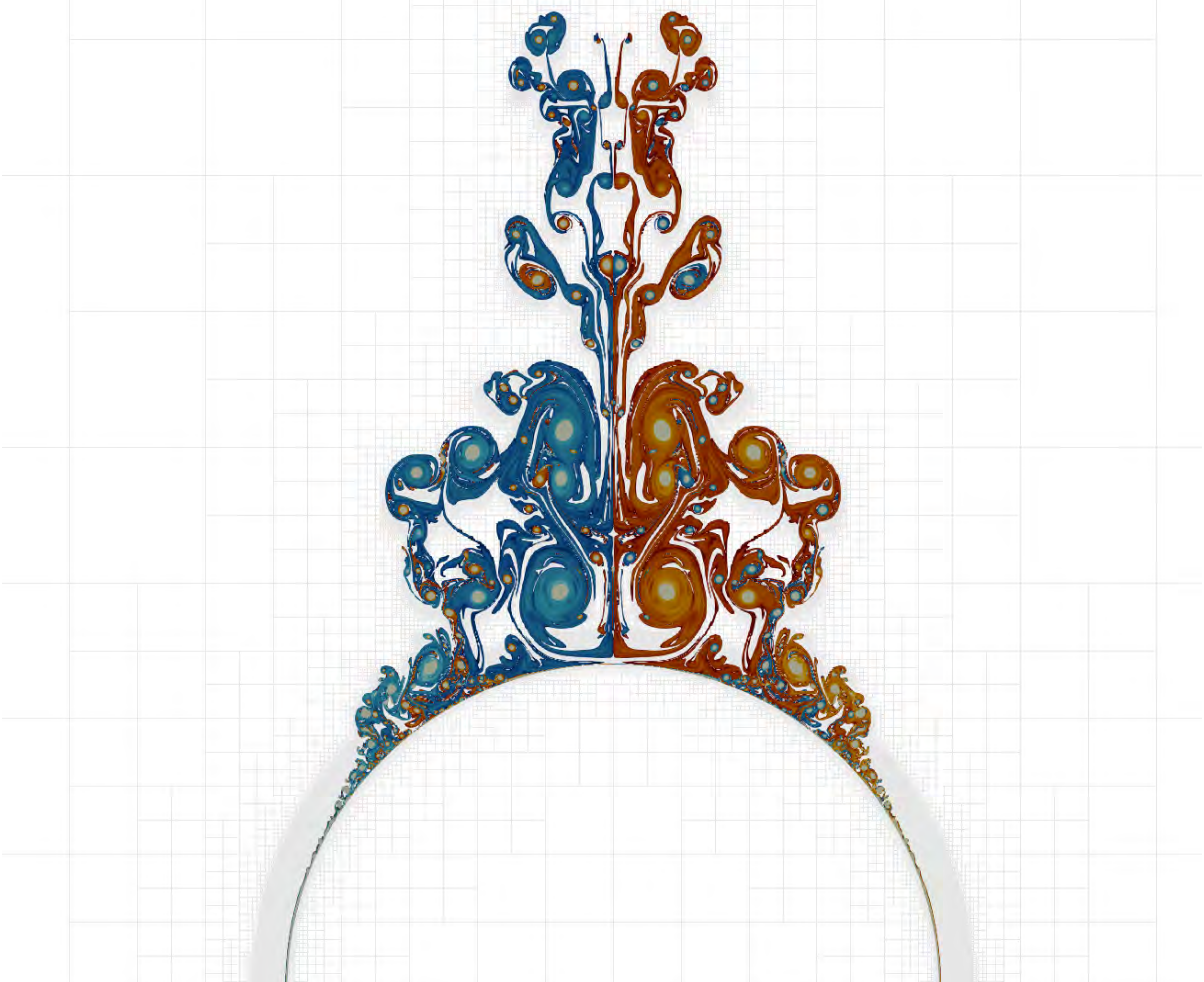
Petr Karnakov[1,a], Sergey Litvinov[1,2,b], and Petros Koumoutsakos[1,c]

[1] Computational Science and Engineering Laboratory, Harvard John A. Paulson School of Engineering and Applied Sciences, 29 Oxford St, Cambridge, MA 02138, USA
[2] Computational Science and Engineering Laboratory, ETH Zurich, Clausiusstrasse 33, 8092 Zurich, Switzerland

2D - Impulsively Started Cylinder



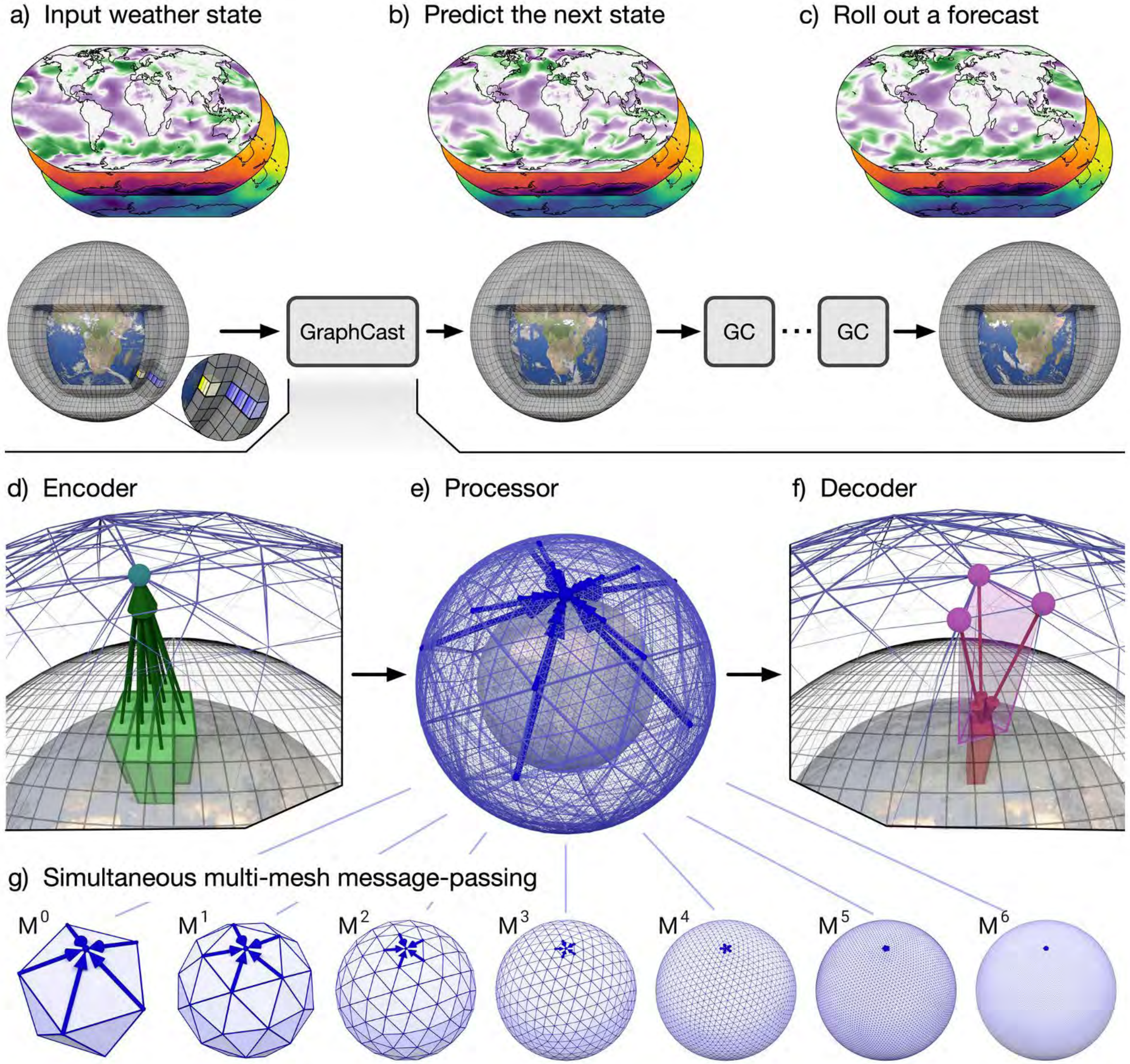
Chatzimanolakis, Weber, Koumoutsakos, JFM, 953, 2022



RESEARCH ARTICLE | WEATHER FORECASTING

Learning skillful medium-range global weather forecasting

REMI LAM, ALVARO SANCHEZ-GONZALEZ, MATTHEW WILLSON, PETER WIRNSBERGER, MEIRE FORTUNATO, FERRAN ALET, SUMAN RAVURI, TIMO EWALDS, ZACH EATON-ROSEN, [...], AND PETER BATTAGLIA +8 authors Authors Info & Affiliations



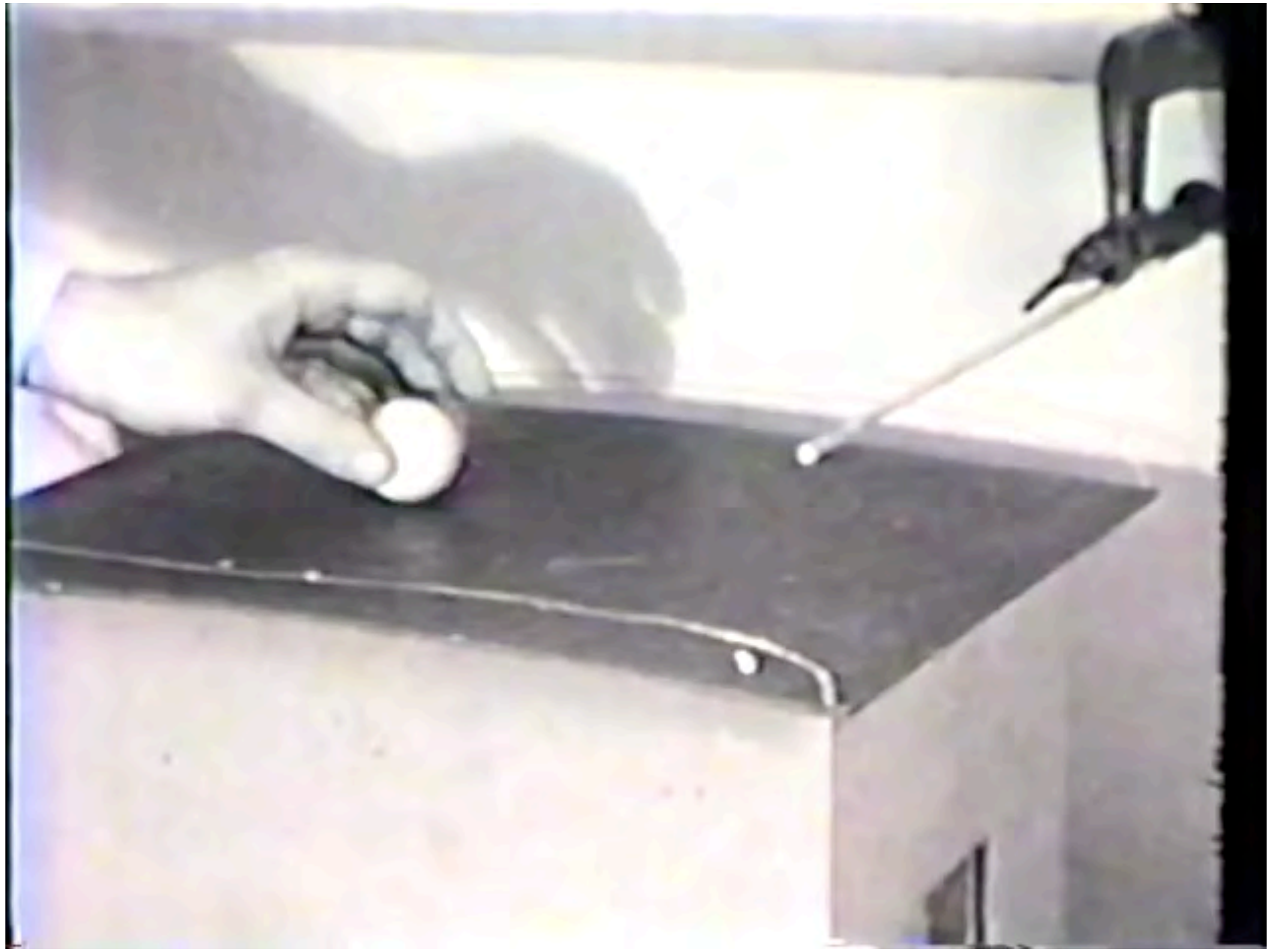
LEARNING TO SOLVE PROBLEMS

ALGORITHMS

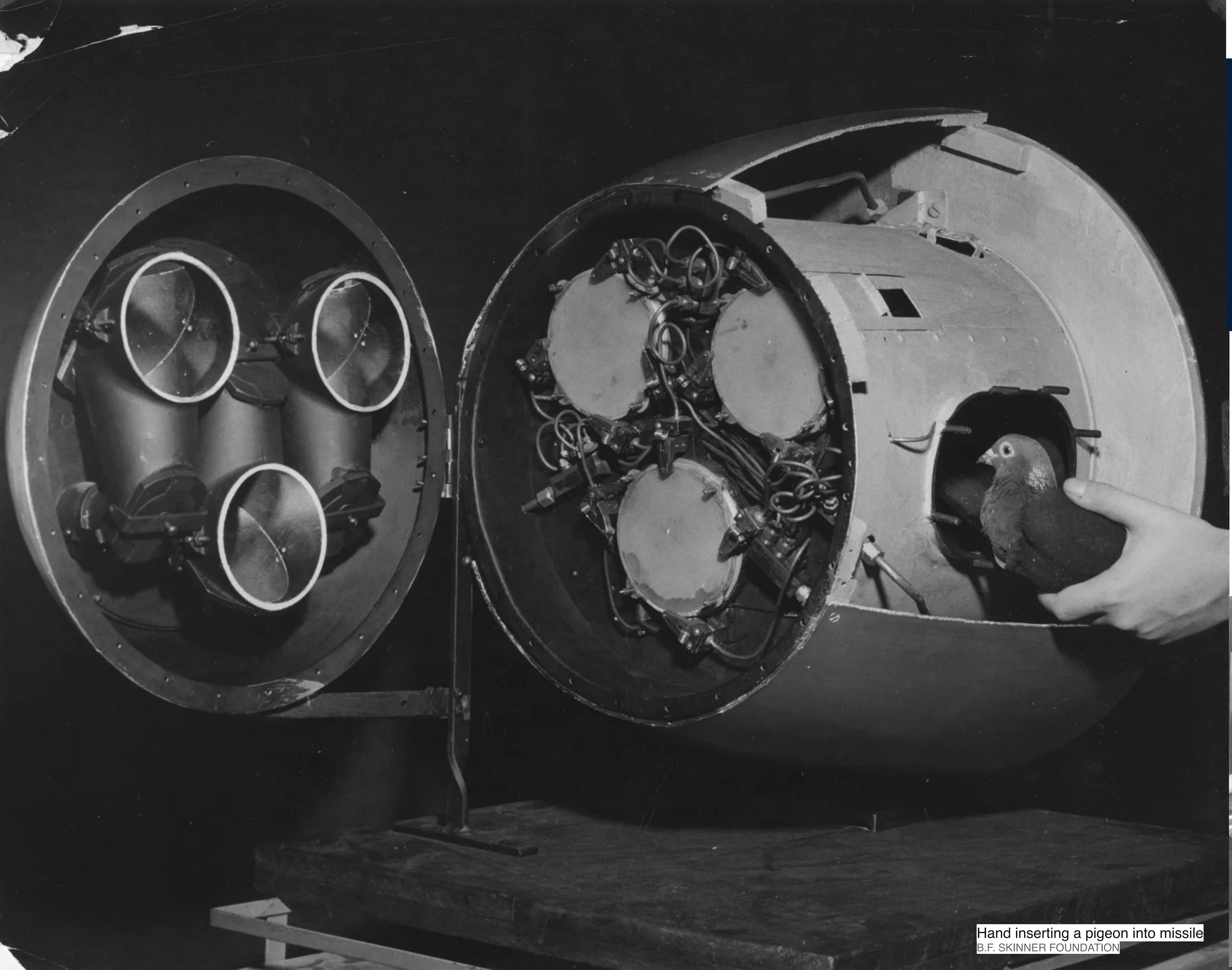
Reinforcement Learning

Learning: Behavioral changes due to Experiences (Action, Stimulus, Reward)

Reinforcement: **stimulus-action** pattern is **rewarded ->** actor is conditioned to a behavior.



CREDIT: B.F. Skinner Foundation



Hand inserting a pigeon into missile
B.F. SKINNER FOUNDATION

MIT Technology Review

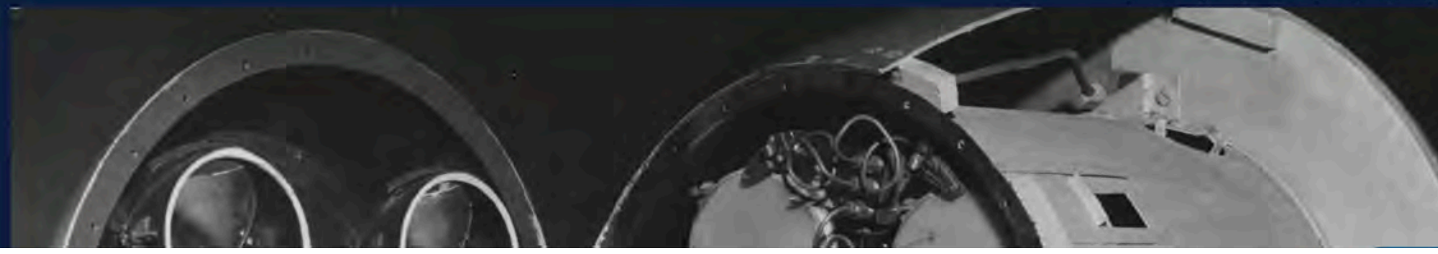
Featured Topics Newsletters Events Podcasts Sign in Subs

1865

Pigeon pilots

B.F. Skinner trained birds to steer bombs. But his prototype was scrapped in favor of a bat-inspired missile guidance system developed at the Rad Lab.

By Christina Couch, SM '15 October 24, 2019

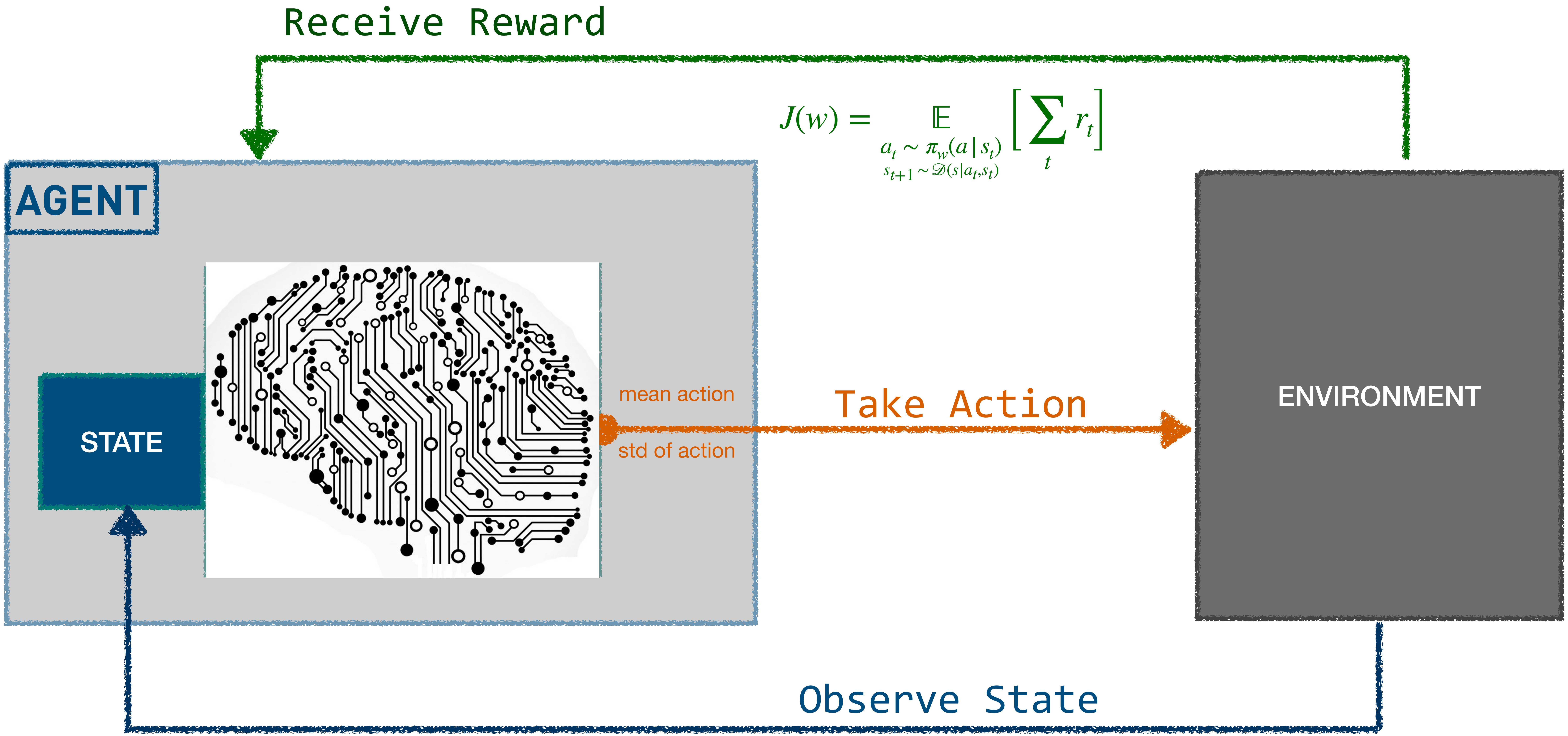


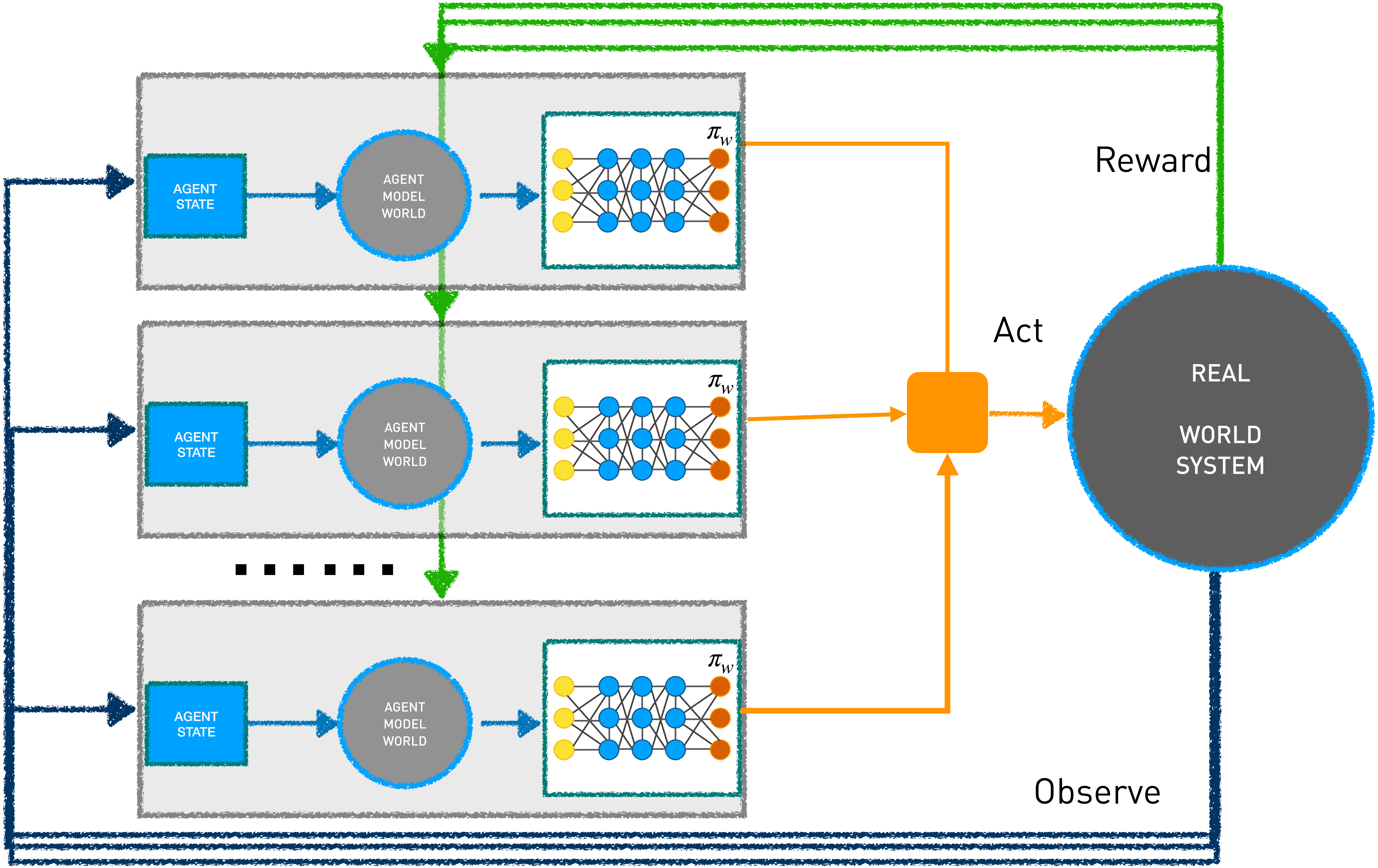
In the early 1940s, as part if the war effort, the Brelands assisted Skinner in his famous Project Pigeon, in which they taught pigeons how to guide bombs. They did this work atop a General Mills grain elevator in Minneapolis, pictured below. Both Keller and Marian left the University of Minnesota without doctorates, planning to apply the powerful procedures they had learned under Skinner to animal behavior. In 1961, when they published their most famous work, they playfully entitled it, The Misbehavior of Organisms.

https://www3.uca.edu/iqzoo/History/bf_skinner.htm

A World Model, from Scott McCloud's *Understanding Comics*. [1, 2]

II. Reinforcement Learning: Find Policy to Maximize Long Term Reward

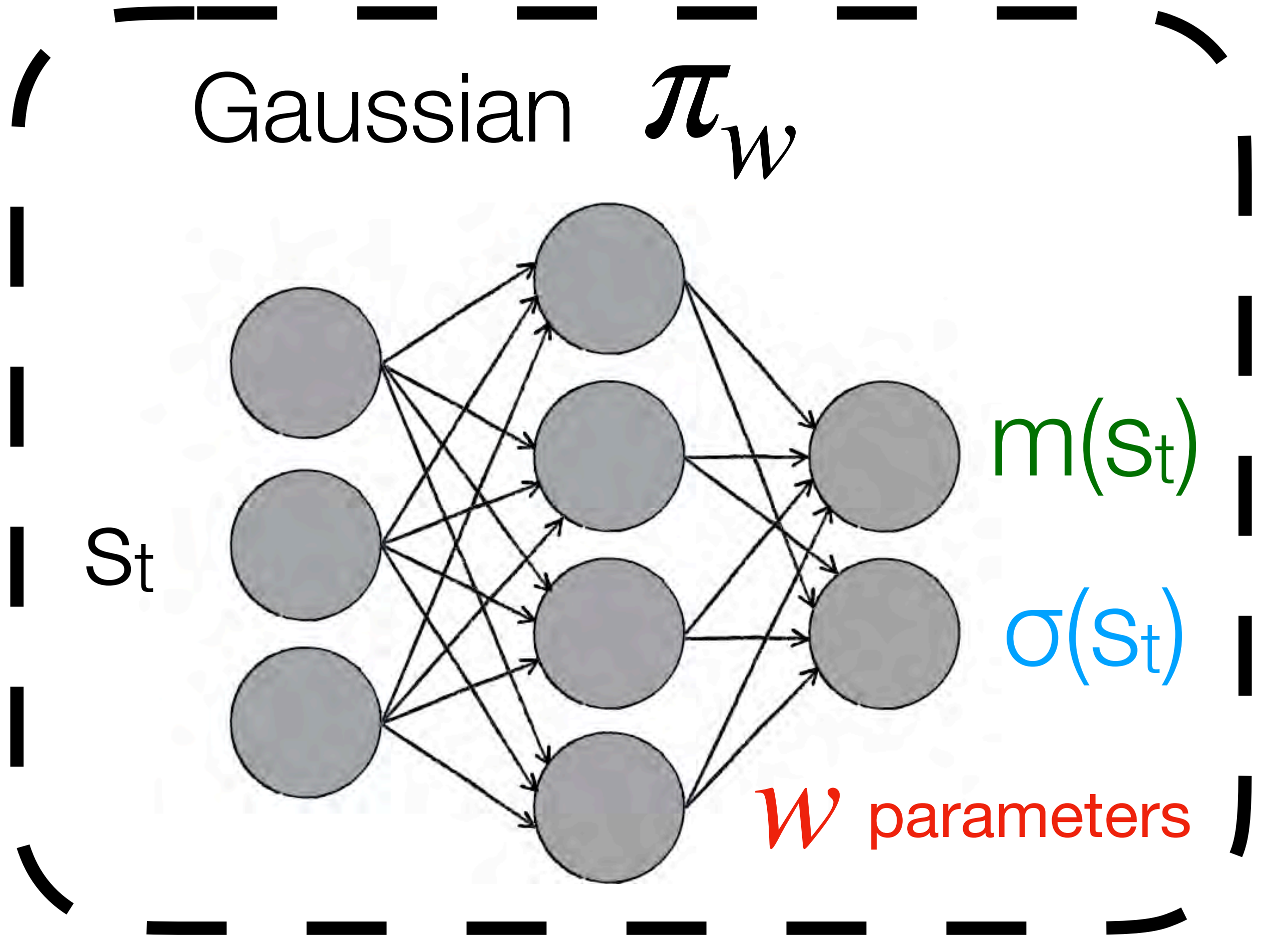




AGENT STATE
AGENT MODEL WORLD
π_w
AGENT STATE
AGENT MODEL WORLD
π_w
AGENT STATE
AGENT MODEL WORLD
π_w
Reward
Act
REAL WORLD SYSTEM
Observe

Policy Iteration Algorithms

- Iteratively improve parameters w of policy network π_w
  - Randomly initialize w
  - Iteratively update w to maximize rewards according to the objective:



$$J(w) = \mathop{\mathbb{E}}_{\substack{a_t \sim \pi_w(a \mid s_t) \\ s_{t+1} \sim \mathcal{D}(s|a_t, s_t)}} \left[\sum_t r_t \right]$$

Find the policy that maximises the sum of **cumulative** rewards

  - Problem: estimated by **sampling** - **high variance**

$$w_{t+1} = w_t + \eta \overline{\nabla J(w)}$$ policy-gradient methods

Iterative scheme to change the mean of the policy to increase the likelihood of high rewards

SAMPLING ONLINE POLICY—> EXPENSIVE

(each sample a simulation)

How to reduce the computational cost of RL and its online sampling ?

Economize by using existing samples -> use memory of the system

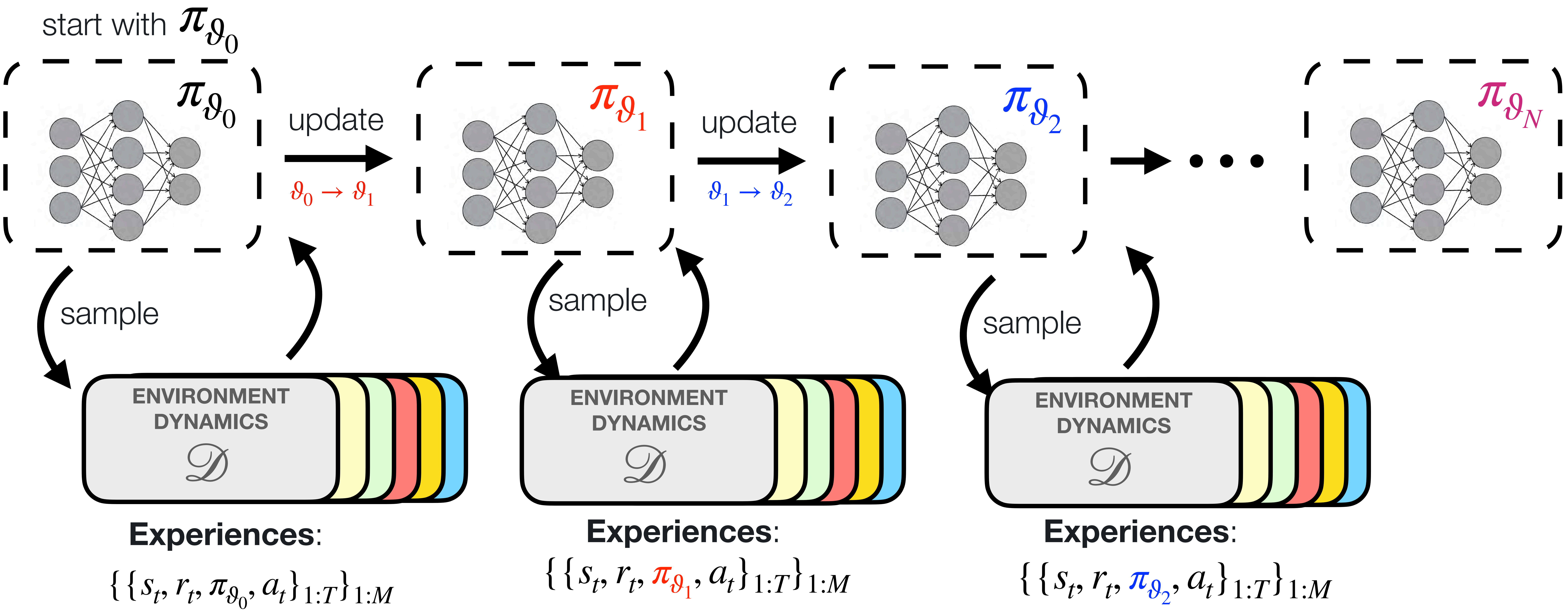
EXPERIENCE REPLAY (Long-Ji Lin, 1992): Store subset of experiences, and "replays" them offline, learning anew from past successes/failures

Experience Replay -> Importance Sampling

Change of probability distribution for computing Expectations

- > BUT how good is that distribution ?

- OFF POLICY LEARNING - EXPERIENCES



After updating ϑ_n, the data collected with policies π_{ϑ_k} $k \in [n-1, n-2, \ldots.]$ are past experiences

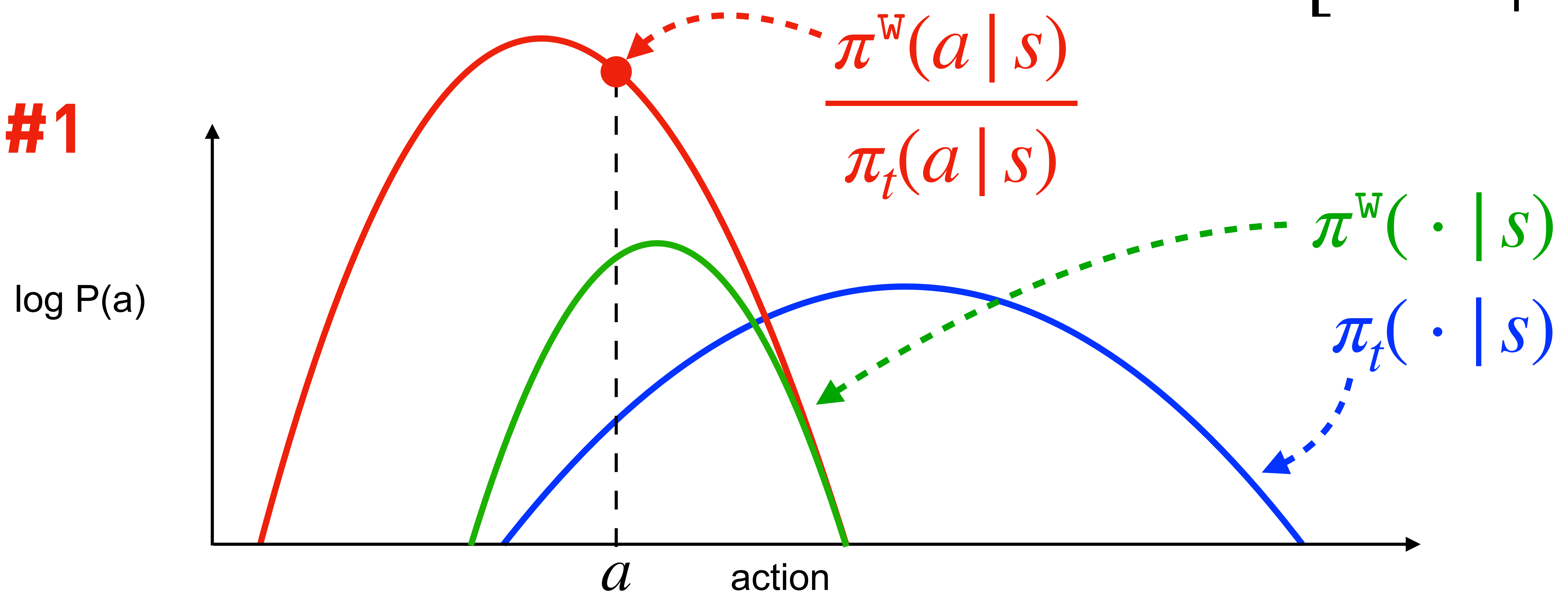
How to utilize off policy/past experiences?

Issues of off-policy RL

Maximize expectation over the policy
by **approximation** on off-policy data

$$J(\mathtt{w}) = \mathbb{E}\left[\hat{J}(s,a)\,\middle|\,\begin{matrix} s \sim \eta^{\pi_{\mathtt{w}}}(\,\cdot\,) \\ a \sim \pi_{\mathtt{w}}(\,\cdot\,|\,s)\end{matrix}\right] \approx \mathbb{E}\left[\frac{\pi_{\mathtt{w}}(a\,|\,s)}{\pi_t(a\,|\,s)}\hat{J}(s,a)\,\middle|\,\begin{matrix} s \sim \eta^{\pi_t}(\,\cdot\,) \\ a \sim \pi_t(\,\cdot\,|\,s)\end{matrix}\right]$$

ISSUE #1

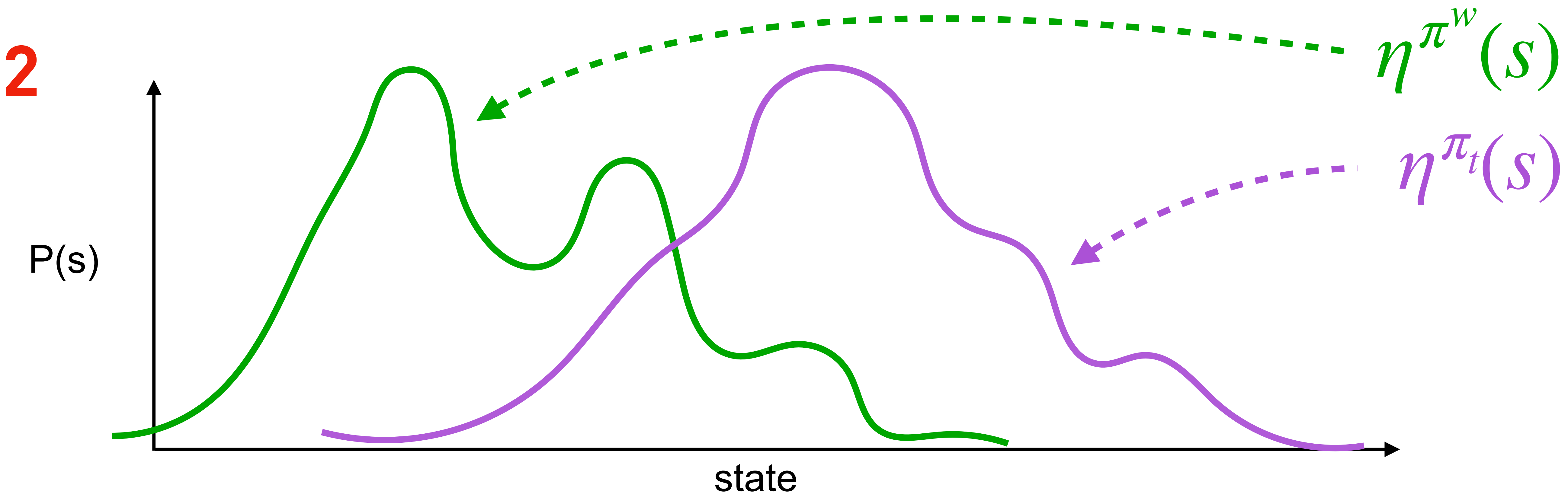


Policy to update

Old policy used by agent in training data

Unbounded (0, ∞) Importance weight: causes **inaccuracy** and **instability** for the learning algorithm.

ISSUE #2



Prob of encountering state with the policy

Prob of encountering state with old policy

If **training data** is too **dissimilar from on-policy** outcomes, data may be **irrelevant to updating the policy**

Remember and Forget Experience Replay

Rule I. Reject samples if importance weight π_w/μ_t lies outside of a ***trust region***.

Rule II. *Penalization, based on* **KL divergence**, attracts policy back towards training behaviors.

(Novati & Koumoutsakos, ICML '19

ReF-ER works with most RL methods that learn a policy by Experience Replay (e.g. DPG, NAF, ACER, SAC, ...)

Sure, let's compare the differences between Remember and Forget Experience Replay (ReF-ER) and Proximal Policy Optimization (PPO) in the context of reinforcement learning (RL).

ReF-ER (Remember and Forget Experience Replay)

Concept: ReF-ER is a novel experience replay method designed to address the issue of catastrophic forgetting in RL. It incorporates mechanisms to remember important experiences and forget less critical ones to maintain a balanced and informative replay buffer.

Key Features:

- **Balanced Experience Replay**: ReF-ER selectively retains experiences that are deemed valuable for learning while discarding those that are less important. This helps in maintaining a diverse and relevant set of experiences for training.
- **Catastrophic Forgetting**: By balancing the experiences, ReF-ER mitigates the problem of catastrophic forgetting, where an agent might forget previously learned behaviors when exposed to new experiences.
- **Prioritization**: Uses a prioritization mechanism to determine which experiences to remember and which to forget. This can be based on factors like the novelty or importance of the experiences.

Advantages:

- **Efficiency**: More efficient use of the replay buffer by retaining only valuable experiences.
- **Improved Learning**: Helps maintain stability in learning by ensuring the agent does not forget important past experiences.
- **Sample Efficiency**: Enhances sample efficiency by focusing on significant experiences, reducing the need for a large number of interactions with the environment.

PPO (Proximal Policy Optimization)

Concept: PPO is a policy gradient method for RL that aims to provide a stable and efficient way to update policies. It achieves this through a clipped surrogate objective that limits the magnitude of policy updates.

Key Features:

- **Policy Gradient**: Directly optimizes the policy by using policy gradients.
- **Clipped Objective Function**: PPO employs a clipped objective to prevent large updates to the policy, which can destabilize learning. This ensures that the updates stay within a certain trust region.
- **Trust Region**: By limiting the policy changes, PPO maintains stability and robustness during training.

Advantages:

- **Stability**: The clipping mechanism provides stable and consistent updates, reducing the risk of performance degradation.
- **Efficiency**: Balances exploration and exploitation effectively, making it efficient in learning complex tasks.
- **Simplicity**: Easier to implement and tune compared to more complex methods like Trust Region Policy Optimization (TRPO), while delivering comparable performance.

Comparison

1. **Approach**:
   - **ReF-ER**: Focuses on experience replay, selectively retaining valuable experiences to improve learning efficiency and mitigate catastrophic forgetting.
   - **PPO**: Uses policy gradients with a clipped objective to ensure stable and efficient policy updates.
2. **Experience Management**:
   - **ReF-ER**: Manages a replay buffer by prioritizing and retaining important experiences, discarding less critical ones.
   - **PPO**: Does not use experience replay; instead, it updates policies using on-policy data with a clipping mechanism for stability.
3. **Stability and Efficiency**:
   - **ReF-ER**: Improves stability by maintaining a diverse and informative set of experiences, reducing the chances of catastrophic forgetting.
   - **PPO**: Achieves stability through the clipped objective function, ensuring that policy updates remain within a safe range.
4. **Application Scenarios**:
   - **ReF-ER**: Particularly useful in environments where maintaining past knowledge is crucial and experiences can be prioritized based on their importance.
   - **PPO**: Versatile and applicable to a wide range of RL problems, especially those requiring stable and efficient policy updates.
5. **Implementation Complexity**:
   - **ReF-ER**: Can be more complex to implement due to the need for a prioritization mechanism and balanced experience management.
   - **PPO**: Relatively straightforward to implement and tune, making it a popular choice for practical applications.

Summary

ReF-ER and PPO address different aspects of the RL problem. ReF-ER enhances the experience replay process to improve learning efficiency and mitigate forgetting, making it suitable for tasks where maintaining past knowledge is critical. PPO, on the other hand, provides a stable and efficient way to update policies through a clipped objective function, making it a versatile and wi applicable RL method. The choice between the two depends on the specific requirements and constraints of the problem at hand.

ReFeR and PPO

- ReF-ER is inspired by the techniques developed for on- policy RL to bound policy changes in PPO (Schulman et al., 2017).

- Rule 1 of ReF-ER is similar to the clipped objective function of PPO (gradients are zero if ρ_t is outside of some range). However, Rule 1 is not affected by the sign of the advantage estimate and clips both policy and value gradients.

- Another variant of PPO penalizes $D_{KL}(\mu_t \| \pi_w)$ in a similar manner to Rule 2 (also Schulman et al. (2015) and Wang et al. (2017) employ trust-region schemes in the on- and off-policy setting respectively). PPO picks one of the two techniques, and the authors find that gradient-clipping per- forms better than penalization.

- Conversely, in ReF-ER Rules 1 and 2 complement each other and can be applied to most off-policy RL methods with parametric policies.

arXiv.org > cs > arXiv:1502.05477

Computer Science > Machine Learning

Trust Region Policy Optimization

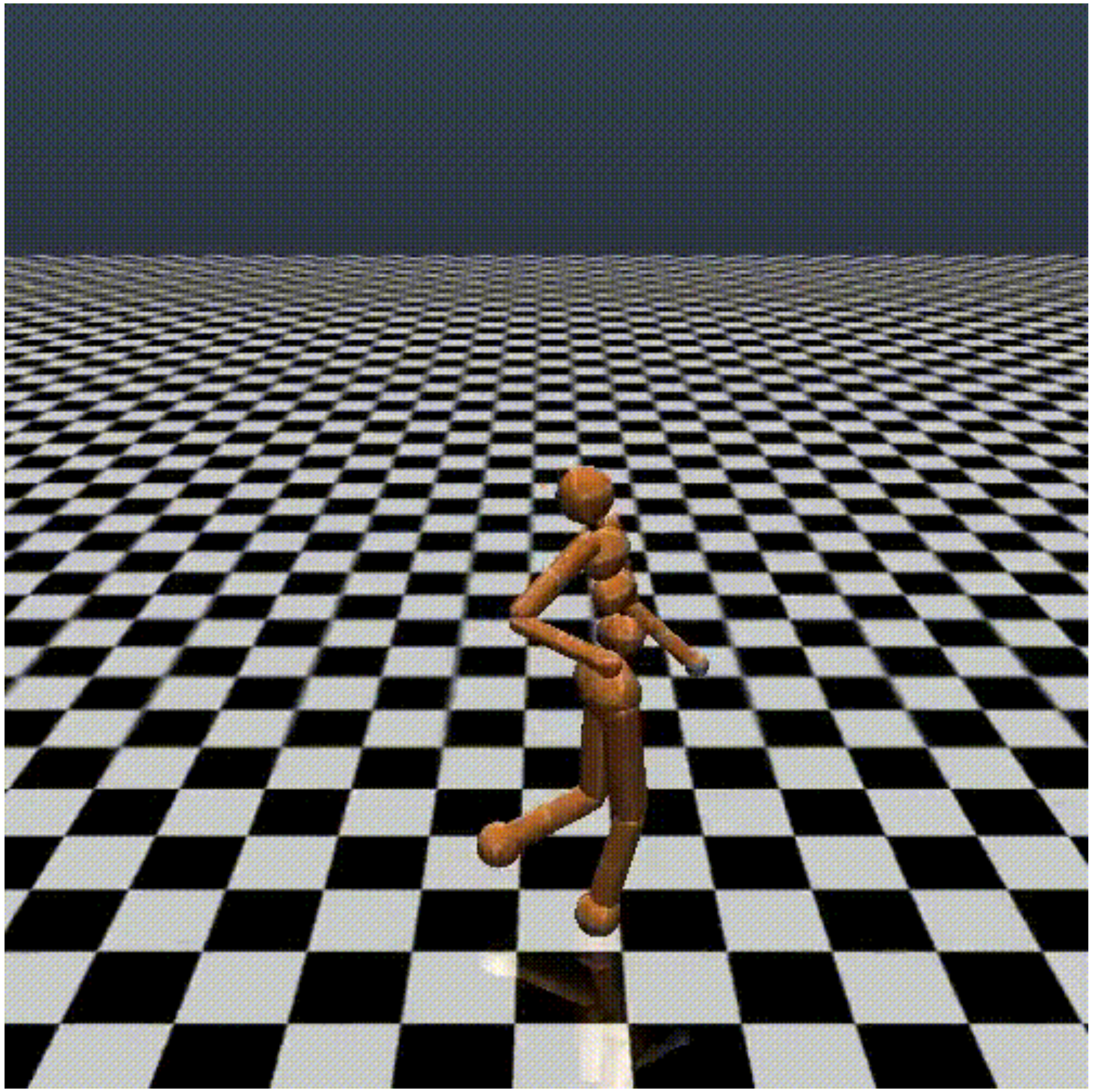
John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, Pieter Abbeel

arXiv.org > cs > arXiv:1807.05827

Computer Science > Machine Learning

Remember and Forget for Experience Replay

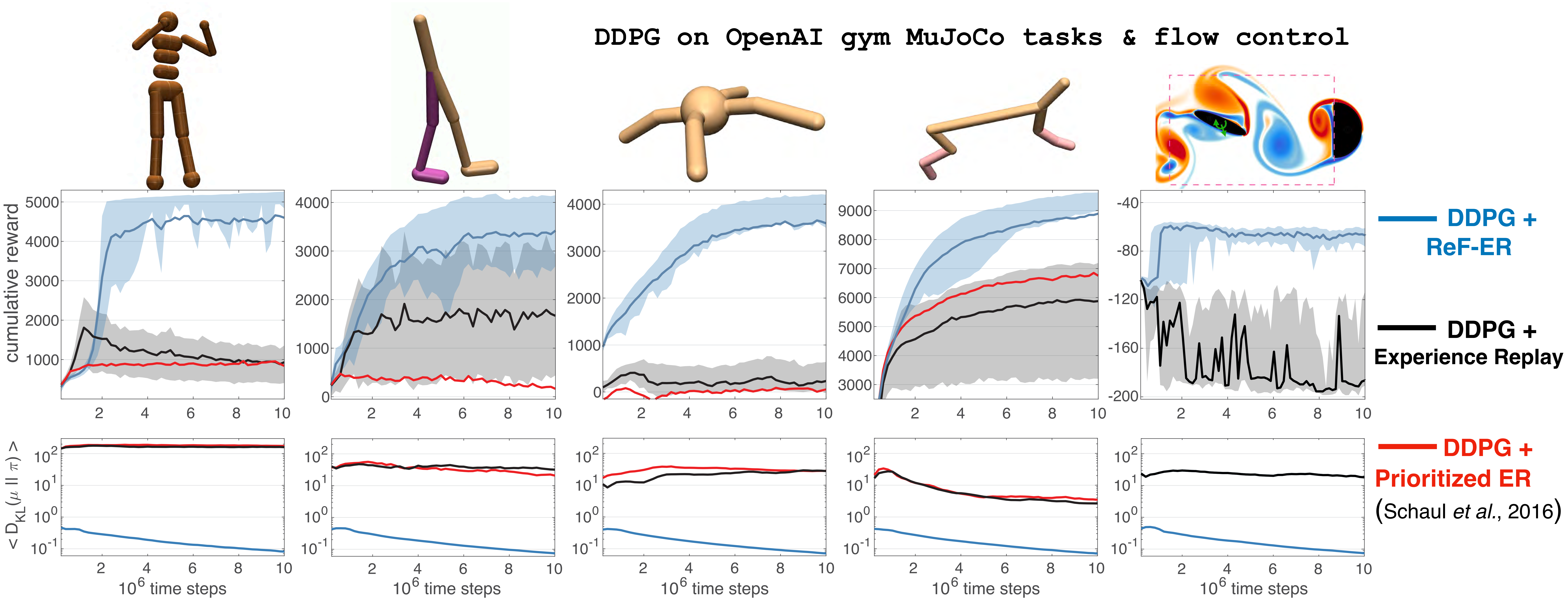
Guido Novati, Petros Koumoutsakos



Humanoid Benchmark

Results

- ReF-ER with: Off-policy pol.-gradients (ACER, Wang *et al.* 2017), Q-learning (NAF, Gu *et al.* 2016), DPG (DDPG, Lillicrap *et al.* 2016).
- We observe: **effectively constrained D_{KL}, increased stability and performance**.
- At the price of: sometimes slower progress at the beginning of training.



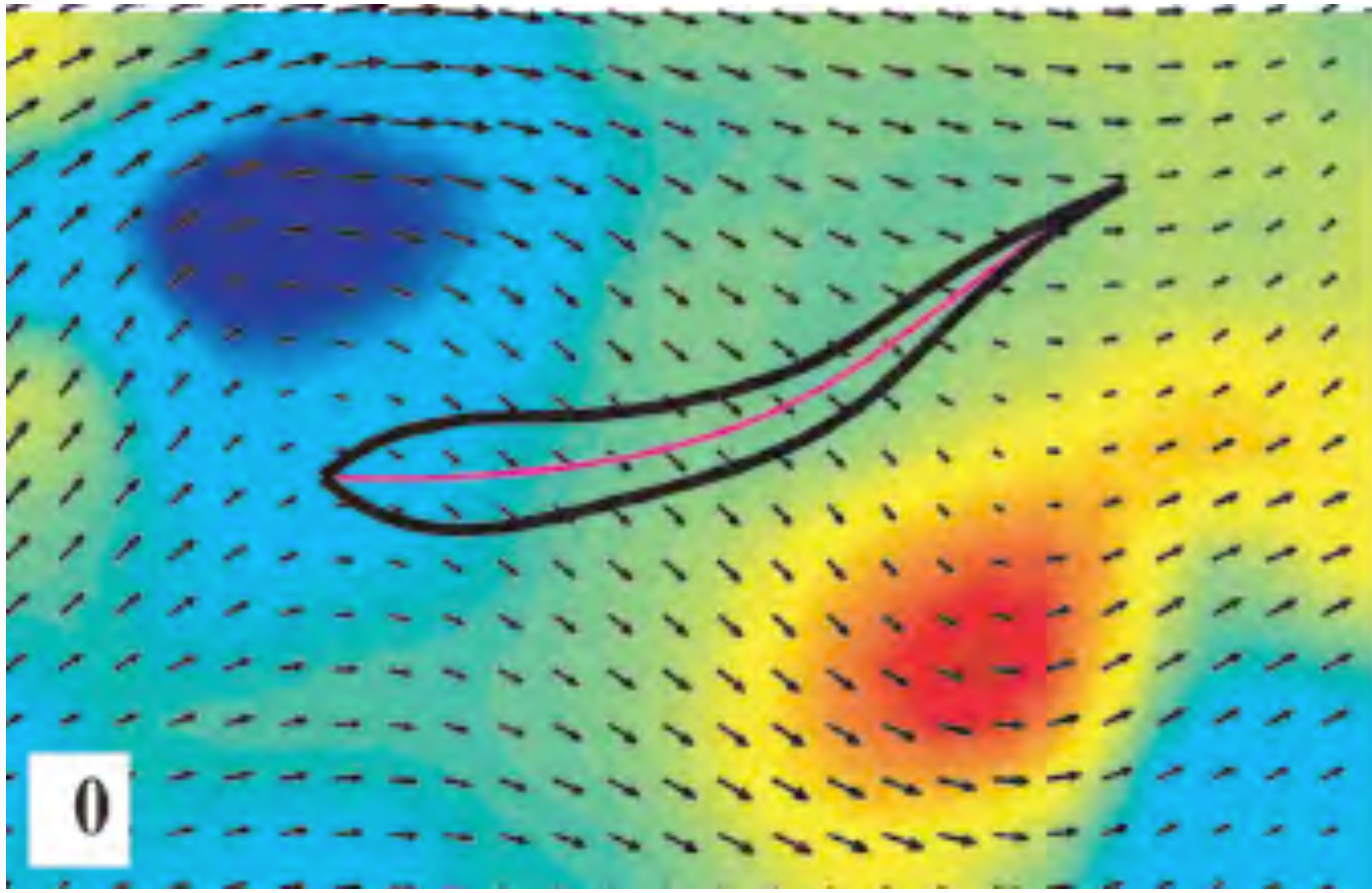
Artificial **Intelligence:** Computational ability to achieve goals

Fish Exploiting Vortices Decrease Muscle Activity

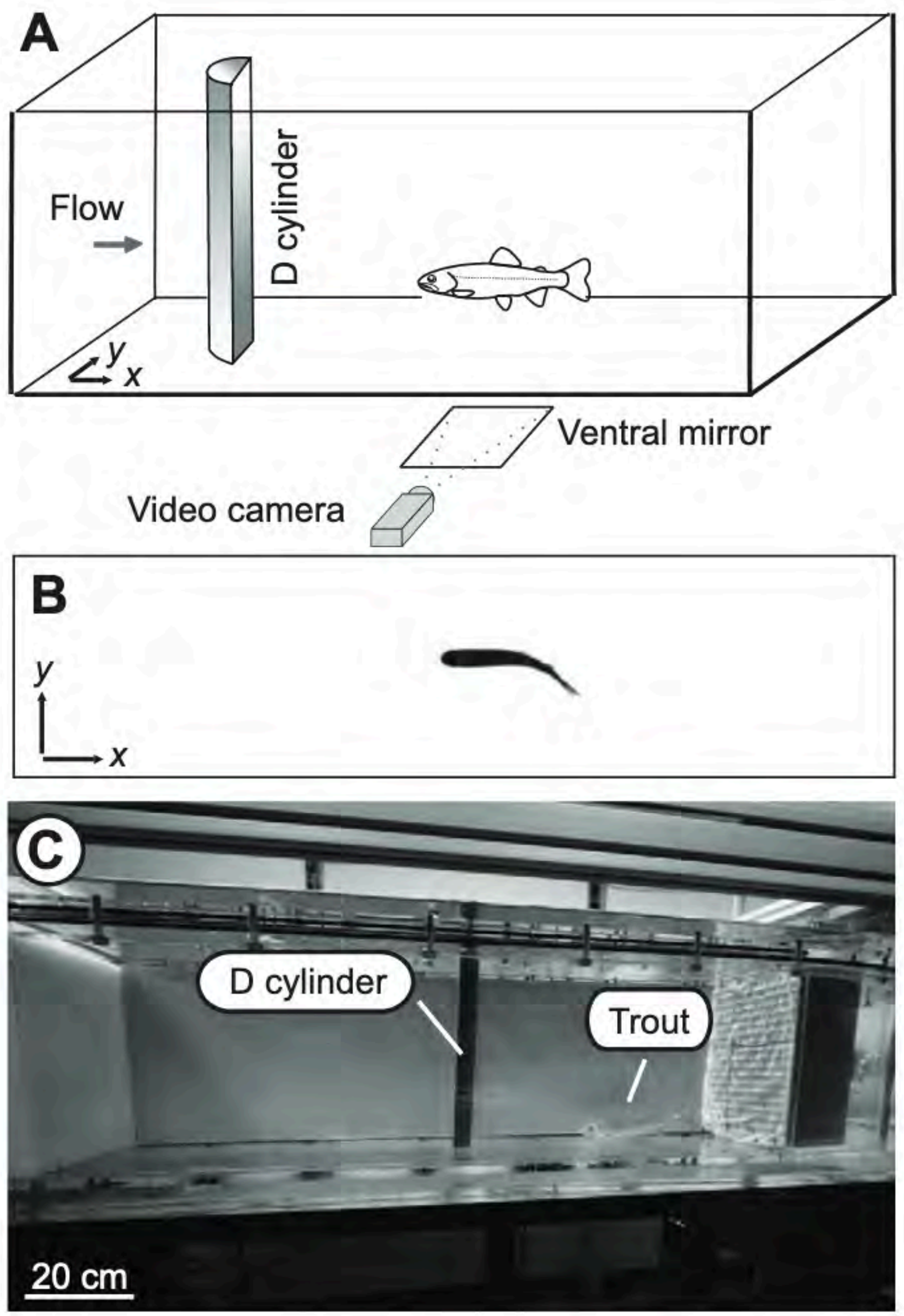
James C. Liao[1,*], David N. Beal[2], George V. Lauder[1], Michael S. Triantafyllou[2]

+ See all authors and affiliations

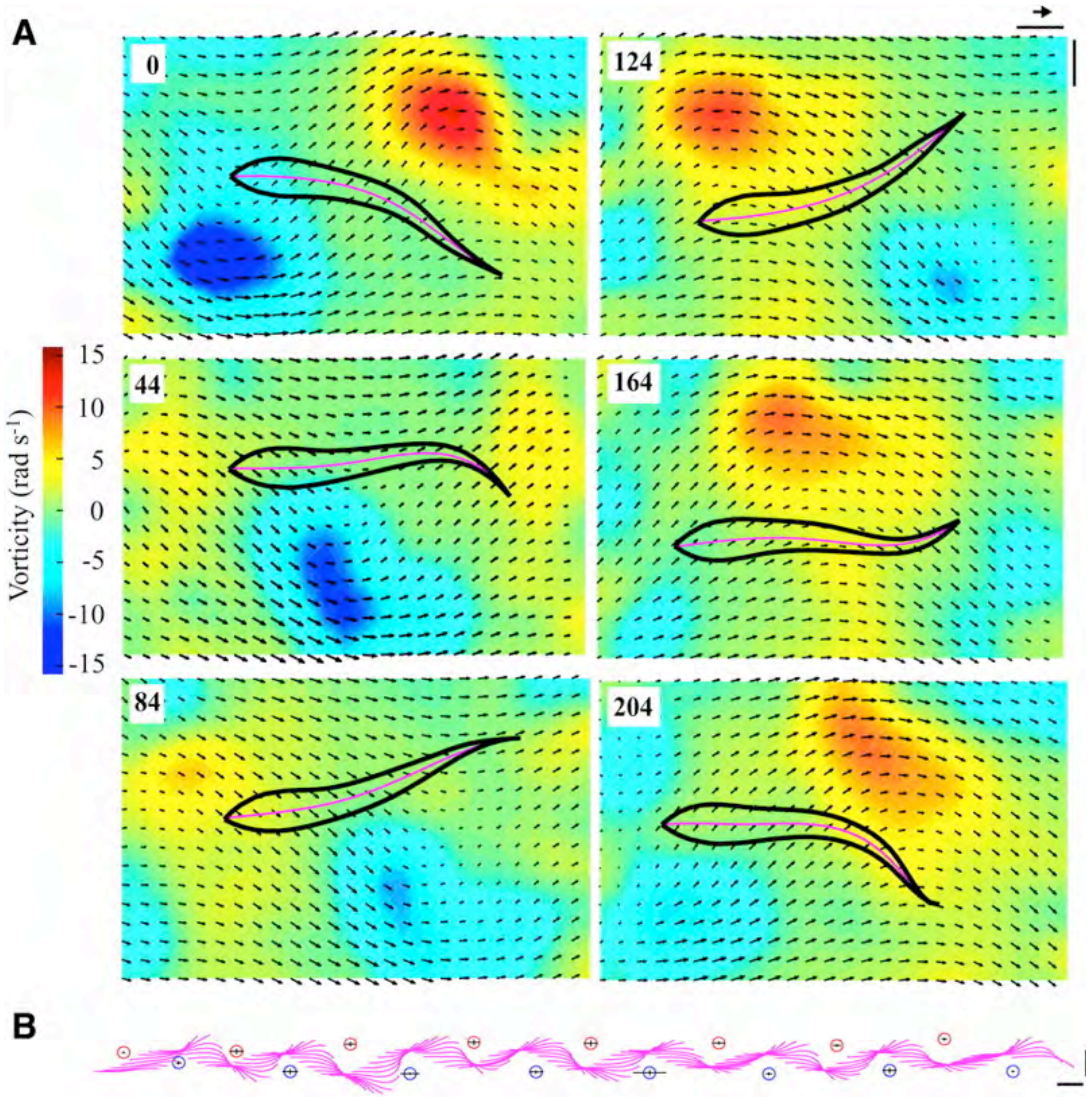
Science 28 Nov 2003:
Vol. 302, Issue 5650, pp. 1566-1569
DOI: 10.1126/science.1088295



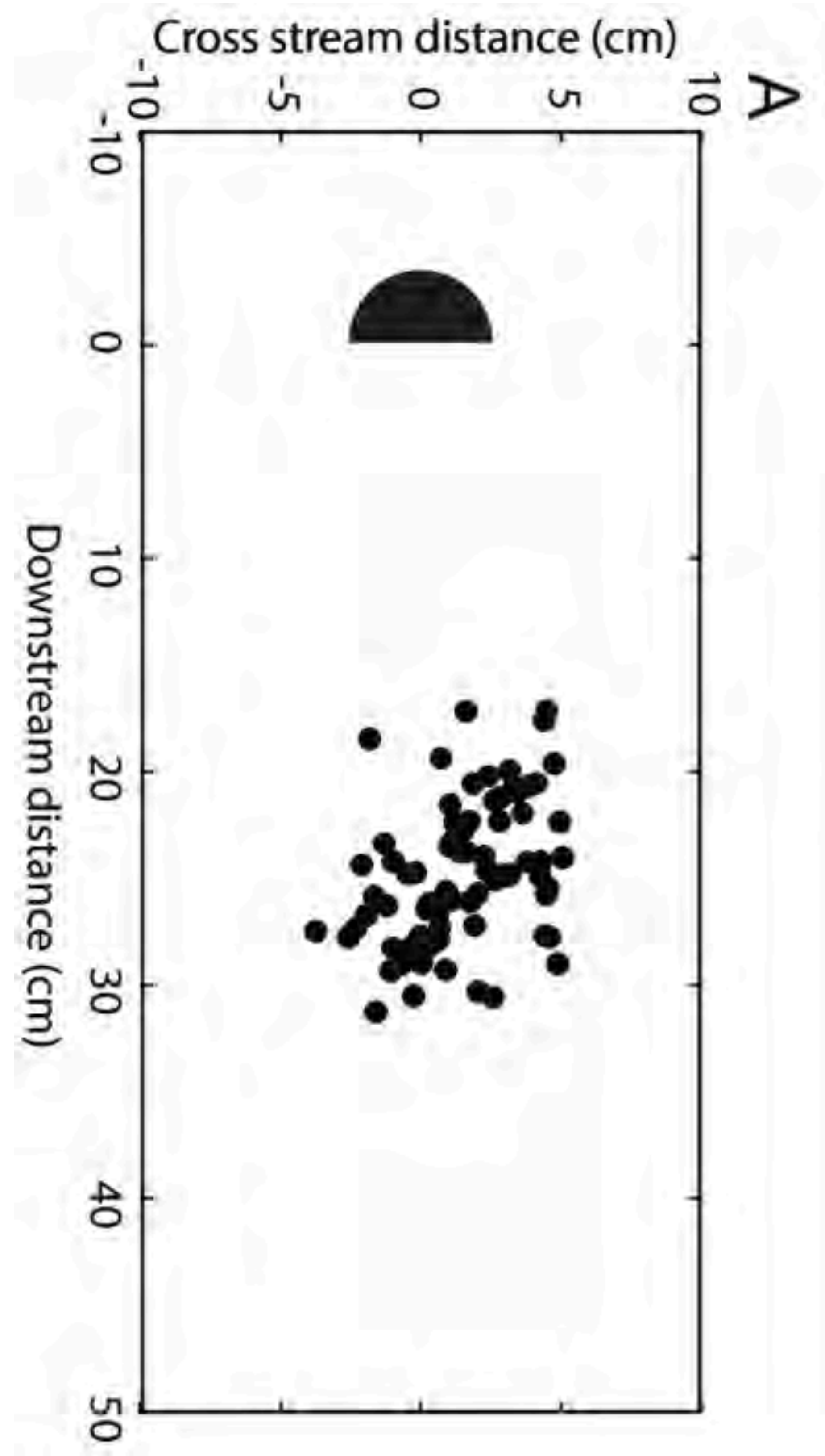
Experiments



Fish slalom between vortices



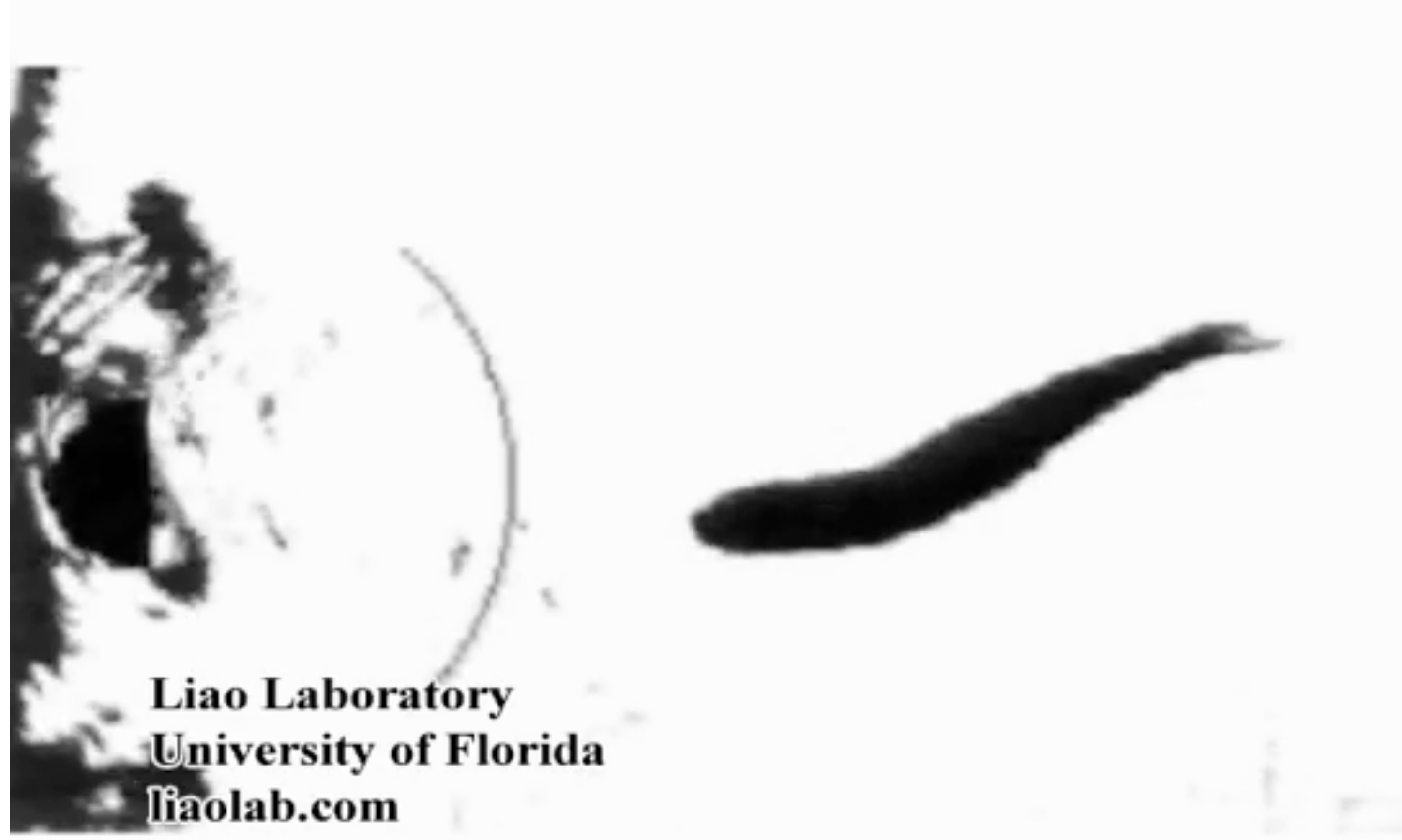
Fish prefer certain regions

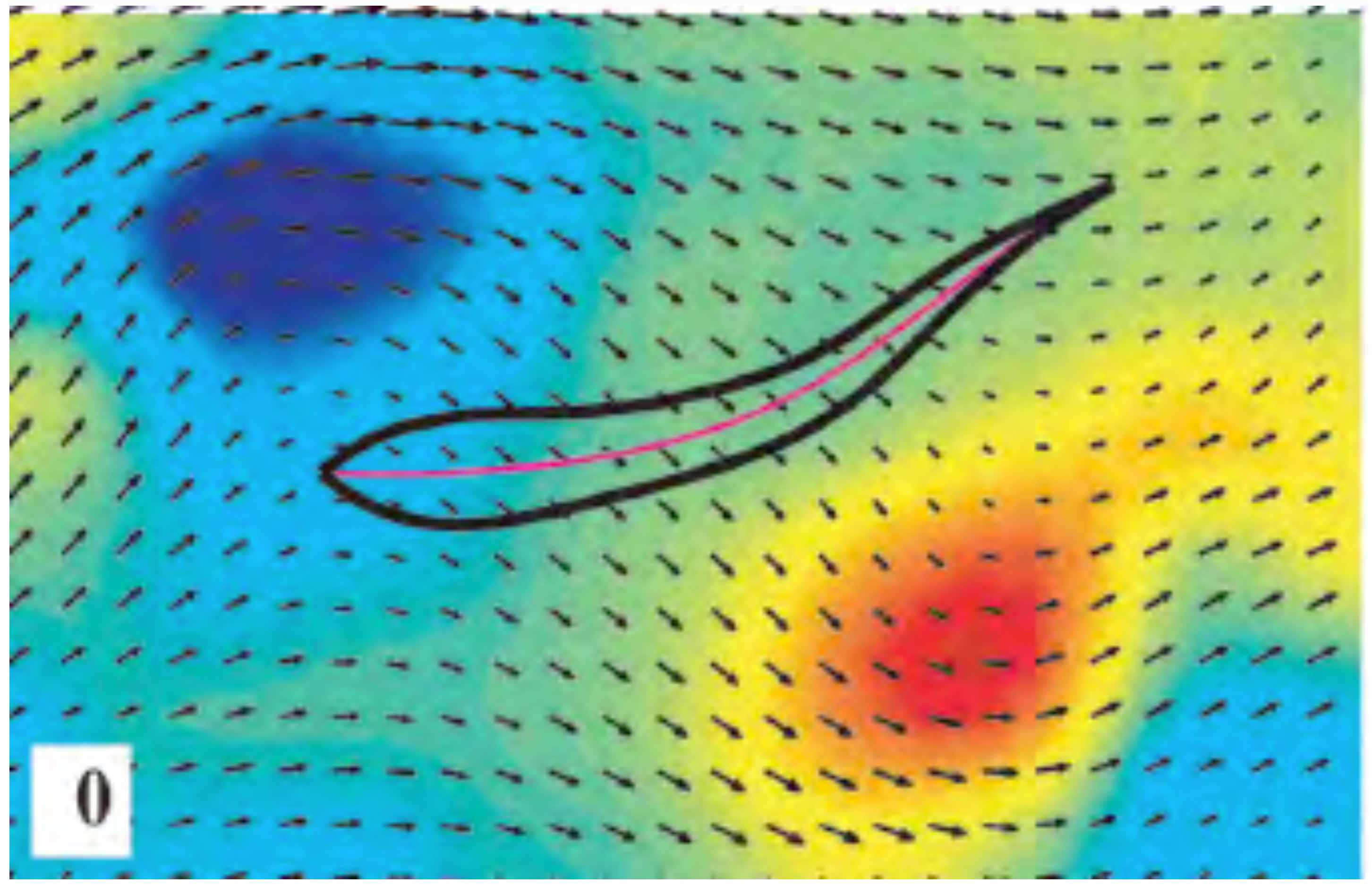


James C. Liao et al., The Kármán gait: novel body kinematics of rainbow trout swimming in a vortex street. J Exp Biol 2003 (10.1242/jeb.00209)
James C. Liao et al., Fish Exploiting Vortices Decrease Muscle Activity, Science 2003 (10.1126/science.1088295)
Otar Akanyeti and James C. Liao; The effect of flow speed and body size on Kármán gait kinematics in rainbow trout. J Exp Biol 2013 (10.1242/jeb.087502)
James C. Liao and Otar Akanyeti, Fish Swimming in a Kármán Vortex Street: Kinematics, Sensory Biology and Energetics, Marine Tech. Soc. J. 2017 (10.4031/MTSJ.51.5.8)
V. Muhawenimana et al., Rainbow trout gait synchronisation with pitching aerofoild vortex shedding, 72nd Annual Meeting of the APS Division of Fluid Dynamics (2019)

Learning to Capture Vortices

Kármán gait employed by fish to decrease effort when swimming behind obstacles





Liao, Beal, Lauder, Triantafyllou, Science 2003

Agent : a **self-propelled swimmer**

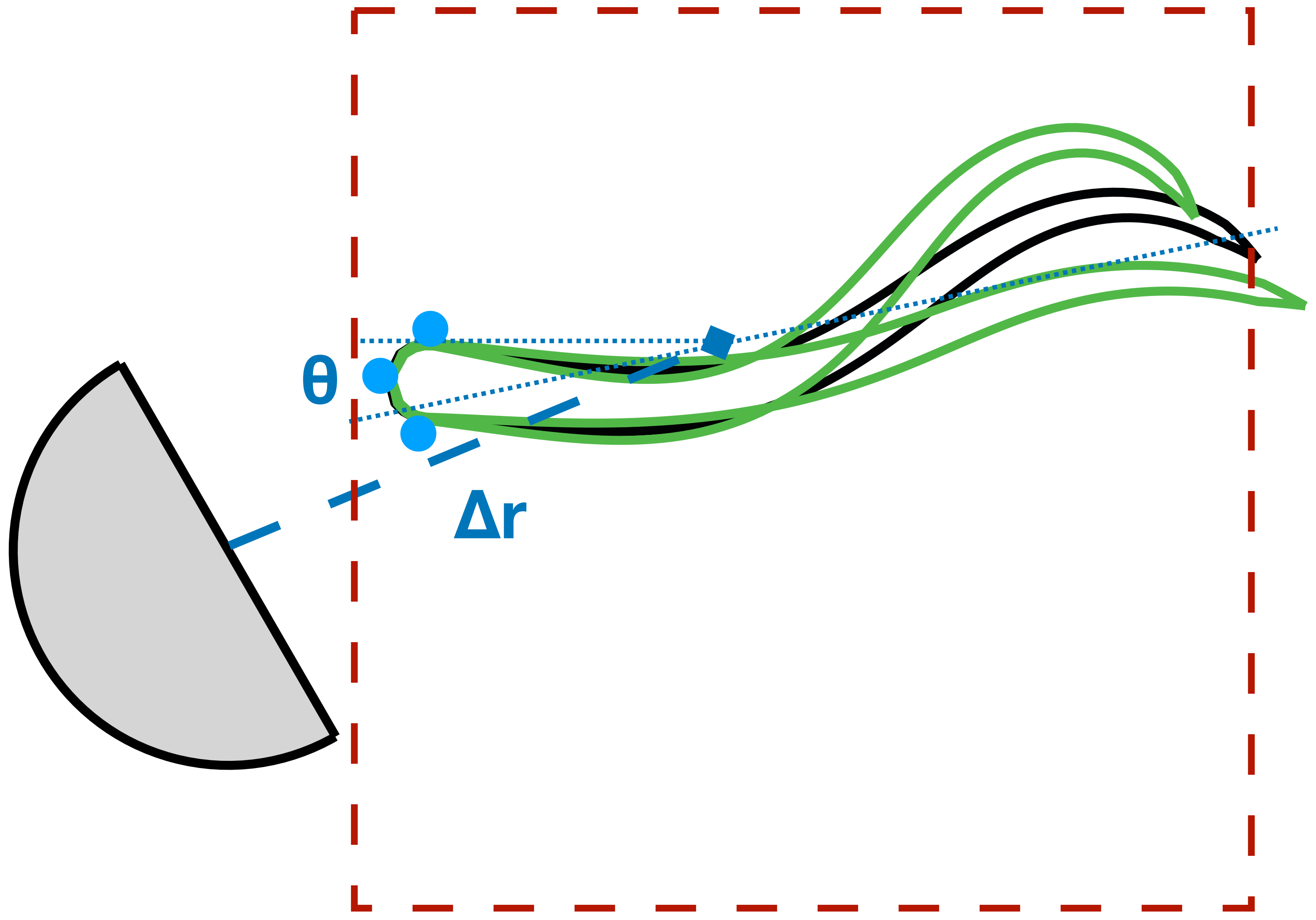
State:
- Relative position **Δr**, angle **θ**, velocities
- Shear stress sensors (lateral-line): ●

Action:
- Increase/decrease undulation amplitude
- Tail-beating frequency

Reward:
- Minimize the swimmer's energy output:

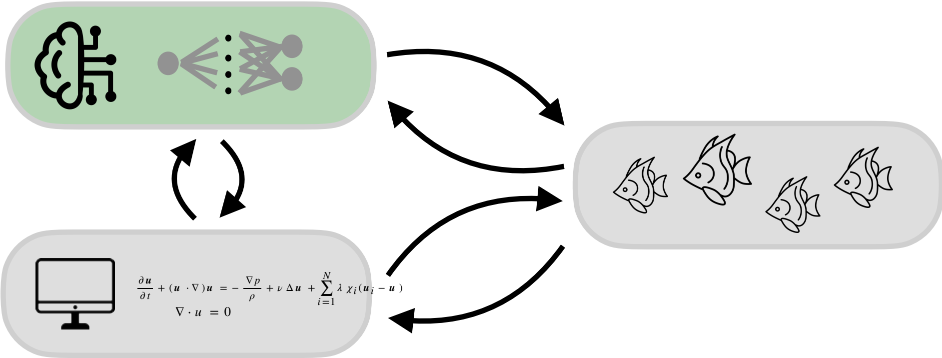
$$r = -P_{\mathrm{def}} = \int_S \mathbf{u}_{\mathrm{deformation}} \cdot \mathrm{d}\mathbf{F}_{\mathrm{fluid}}$$

- Terminate if reaches border: $r_T = -100$



- **By slaloming between vortices, swimmer's power output decreases by 45% relative to swimming in quiescent flow**

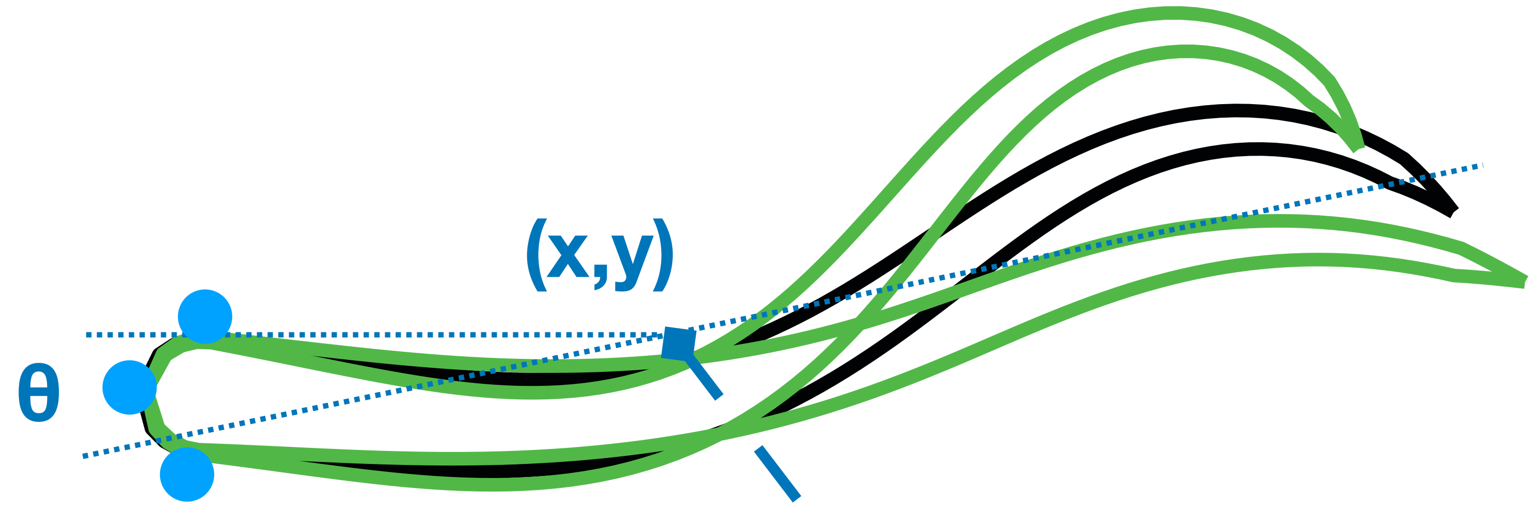
Reinforcement Learning



State: sight: Relative position **(x,y)**, angle **θ**, velocities

lateral line: Shear stress sensors (lateral-line):

proprioception: Shape in previous two timesteps



Reward: efficiency: Swimmer's Froude efficiency $r = \eta \equiv \dfrac{P_{forward}}{P_{forward} + \max(0, P_{deformation})}$

$\left(P_{forward} = \frac{1}{2}\int (\mathbf{u}\cdot d\mathbf{F} + |\mathbf{u}\cdot d\mathbf{F}|)\right.$ and $\left.P_{deformation} = \int \mathbf{u}_{\text{deformation}} d\mathbf{F}\right)$

termination: if leaving domain or touching obstacle $r_T = -10$

G. Novati, et al., Synchronisation through learning for two self-propelled swimmers, *Bioinspir. Biomim.* (2017).

Results - Re=1’000

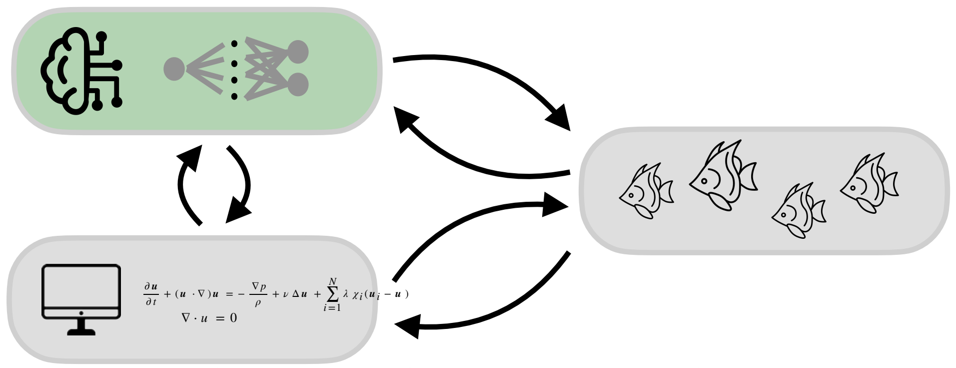
2D Navier-Stokes equation with Brinkman Penalisation

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{\nabla p}{\rho} + \nu \Delta \boldsymbol{u} + \sum_{i=1}^{N} \lambda \chi_i (\boldsymbol{u}_i - \boldsymbol{u})$$

$$\nabla \cdot \boldsymbol{u} = 0$$

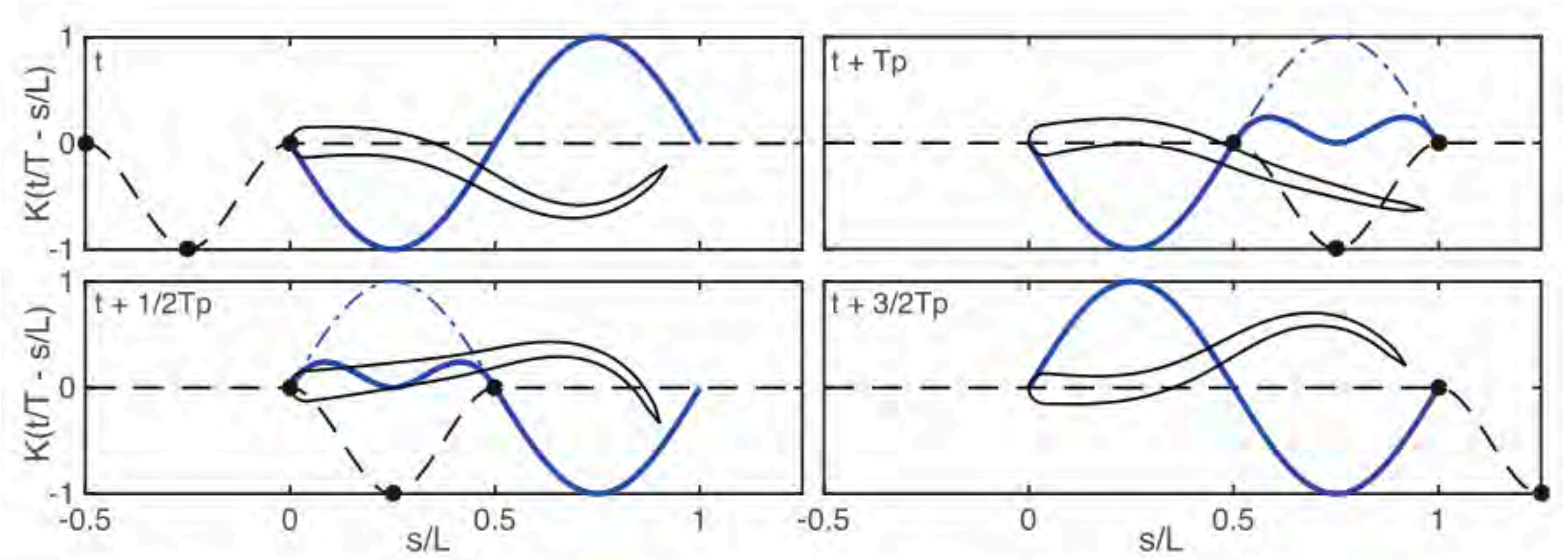
M. Chatzimanolakis, P. Weber, and P. Koumoutsakos, CubismAMR - A C++ library for Distributed Block-Structured Adaptive Mesh Refinement, arxiv e-print (2022).

Reinforcement Learning



Swimmer motion is prescribed via midline curvature

$$k(s,t) = A(s)\left[\sin\left(\frac{2\pi t}{T} - \frac{2\pi s}{L}\right) + \boxed{K\left(\frac{2\pi t}{T} - \frac{2\pi s}{L}\right)}\right]$$

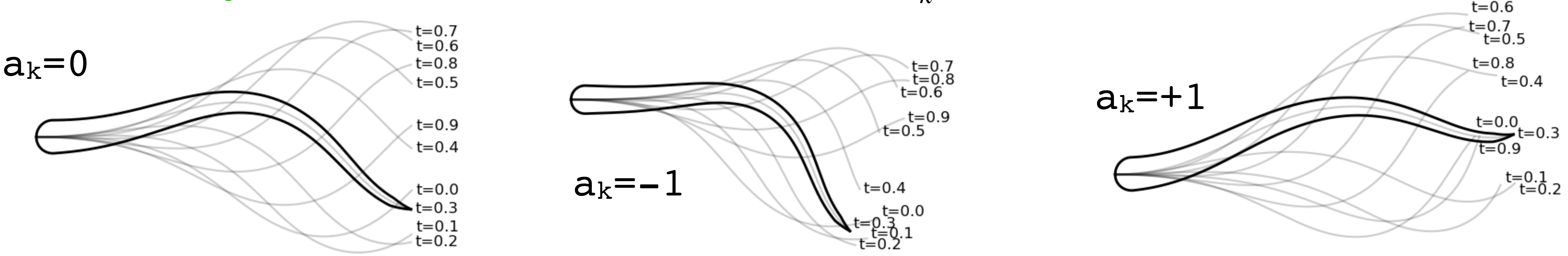


Control inputs a_k modify $K(s,t) = \sum_k a_k m\left(\frac{2\pi t}{T} - \frac{2\pi s}{L}\right)$ with cubic interpolation $m(s,t)$ with

$m(0) = m(1/2) = 0$, $m(1/4) = 1$, and $m'(0) = m'(1/2) = m'(1/4) = 0$. $A(s)$ is linear with $A(0) = 0.82$ and $A(L) = 5.7$

Actions: accelerate: Tail-beating frequency (T)

steering: Increase/decrease midline curvature (a_k)



G. Novati, et al., Synchronisation through learning for two self-propelled swimmers, *Bioinspir. Biomim.* (2017).

Sensitivity/Failure of Deep Reinforcement Learning

Policy trained at $Re = 1000$

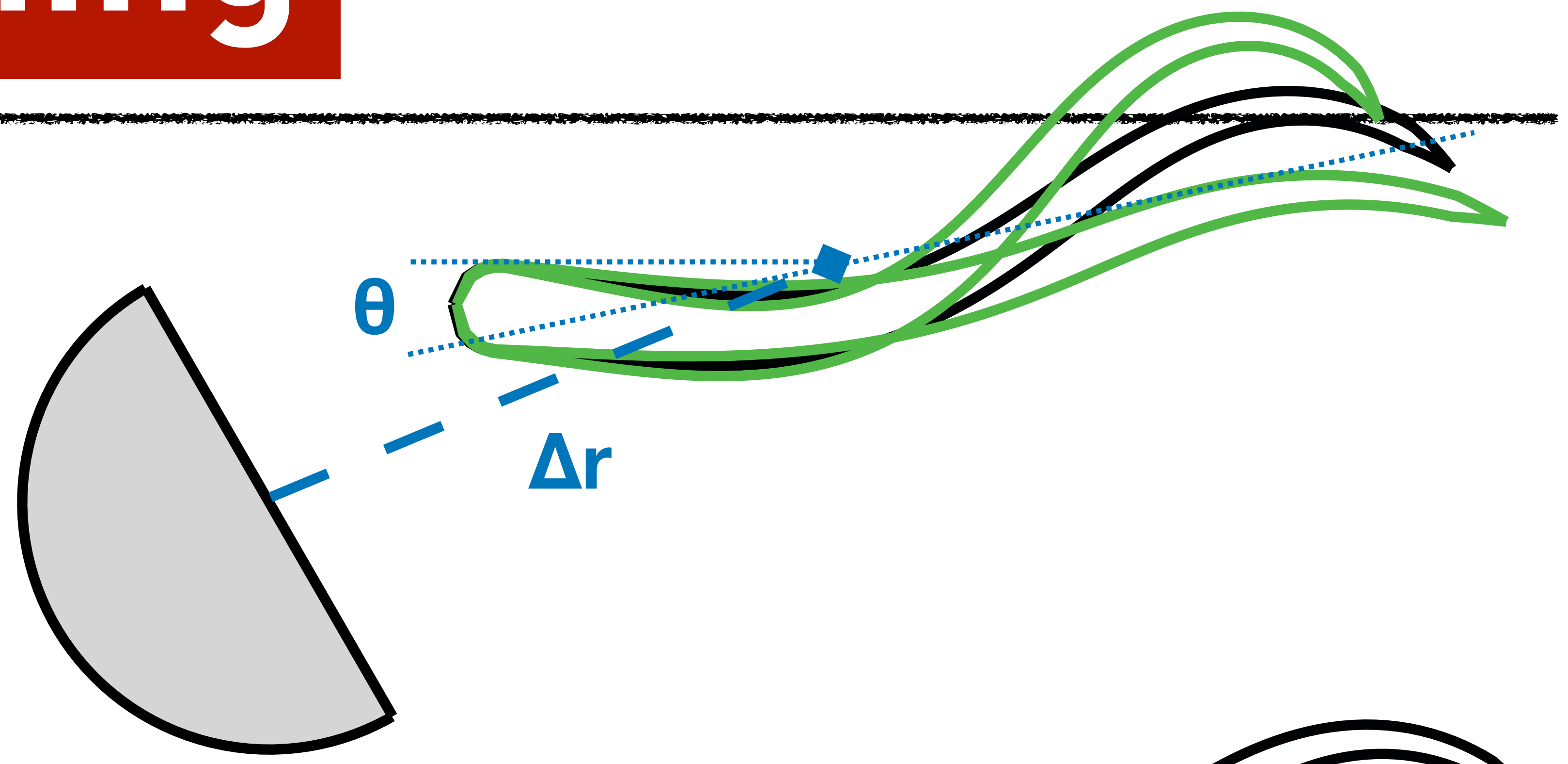
at $Re = 1200$

at $Re = 2000$

FAILURE of Deep Reinforcement Learning

State:

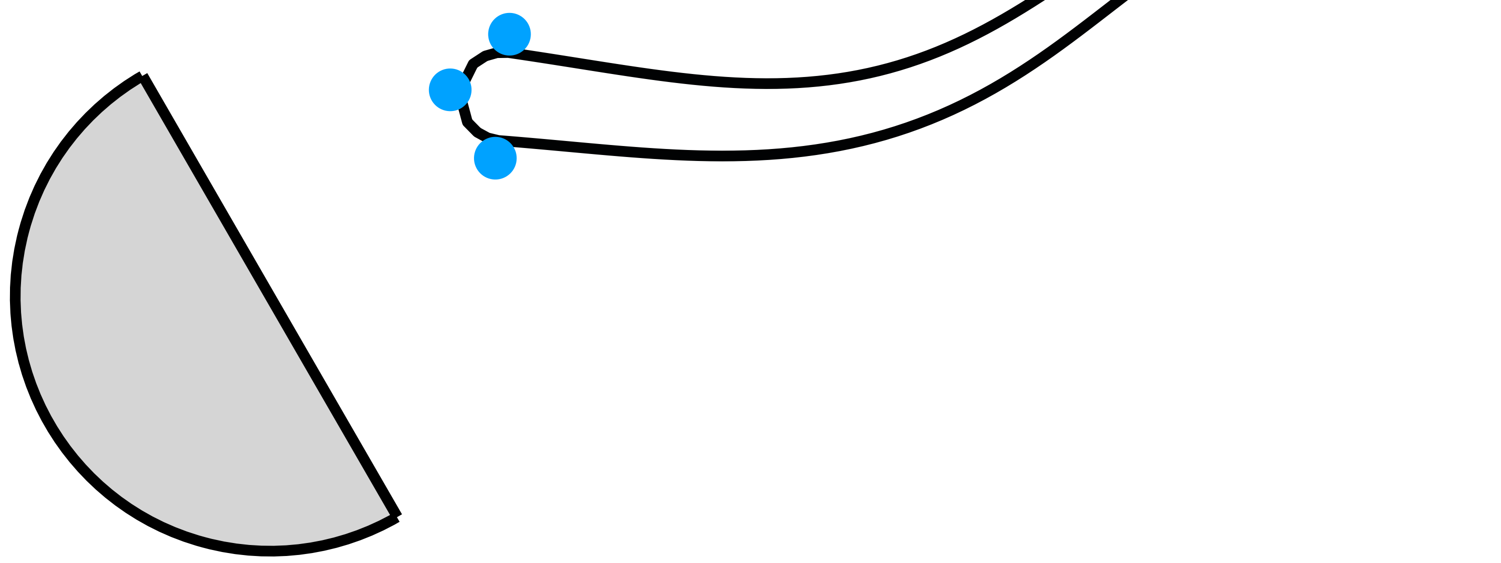
- Relative position **Δr**, angle **θ**, velocities
- Shape in previous two timesteps
- ~~Shear stress sensors (lateral-line):~~



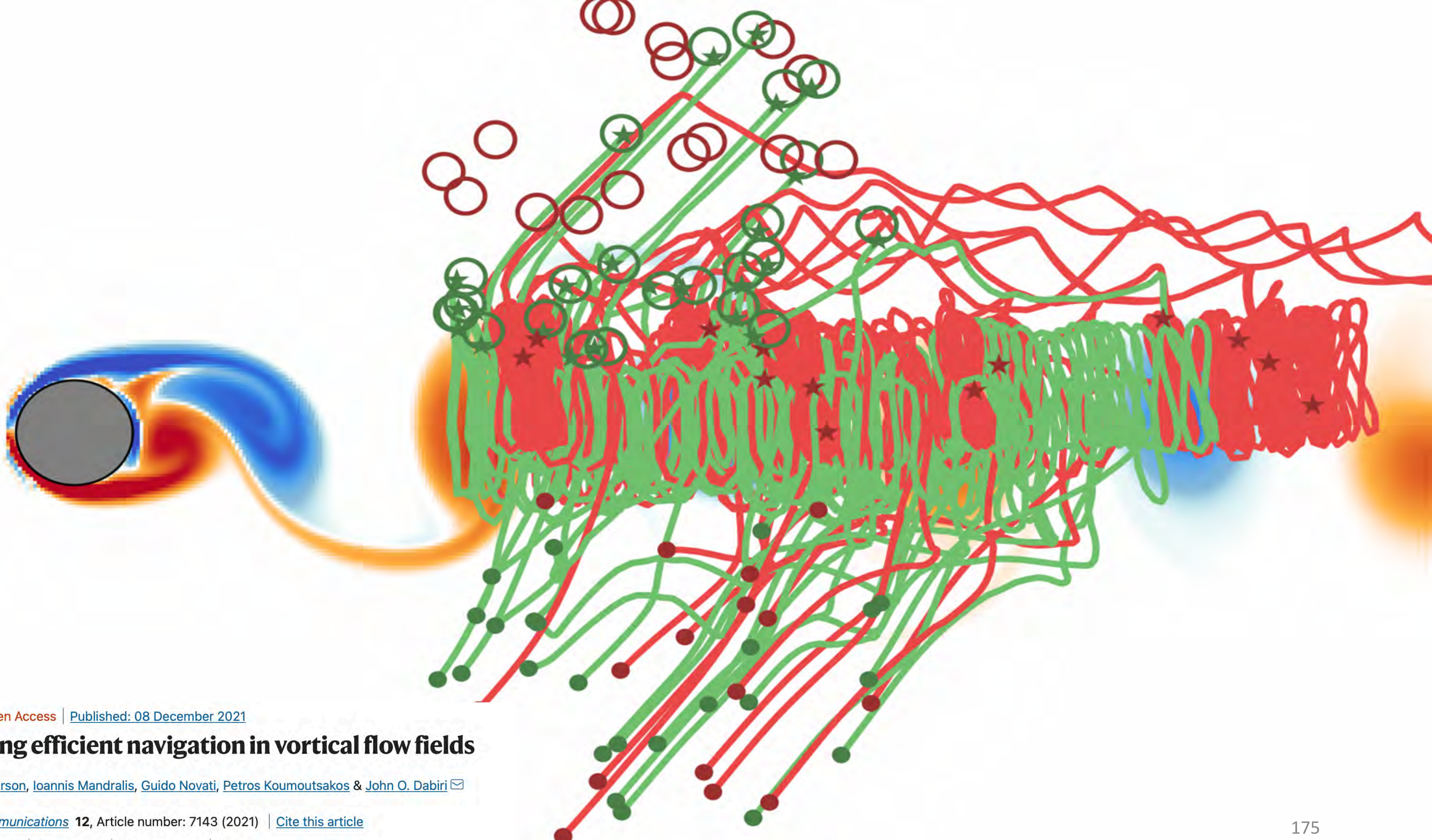
or

State:

- ~~Relative position **Δr**, angle **θ**, velocities~~
- ~~Shape in previous two timesteps~~
- Shear stress sensors (lateral-line):



Deep Reinforcement Learning is very sensitive to the choice of states



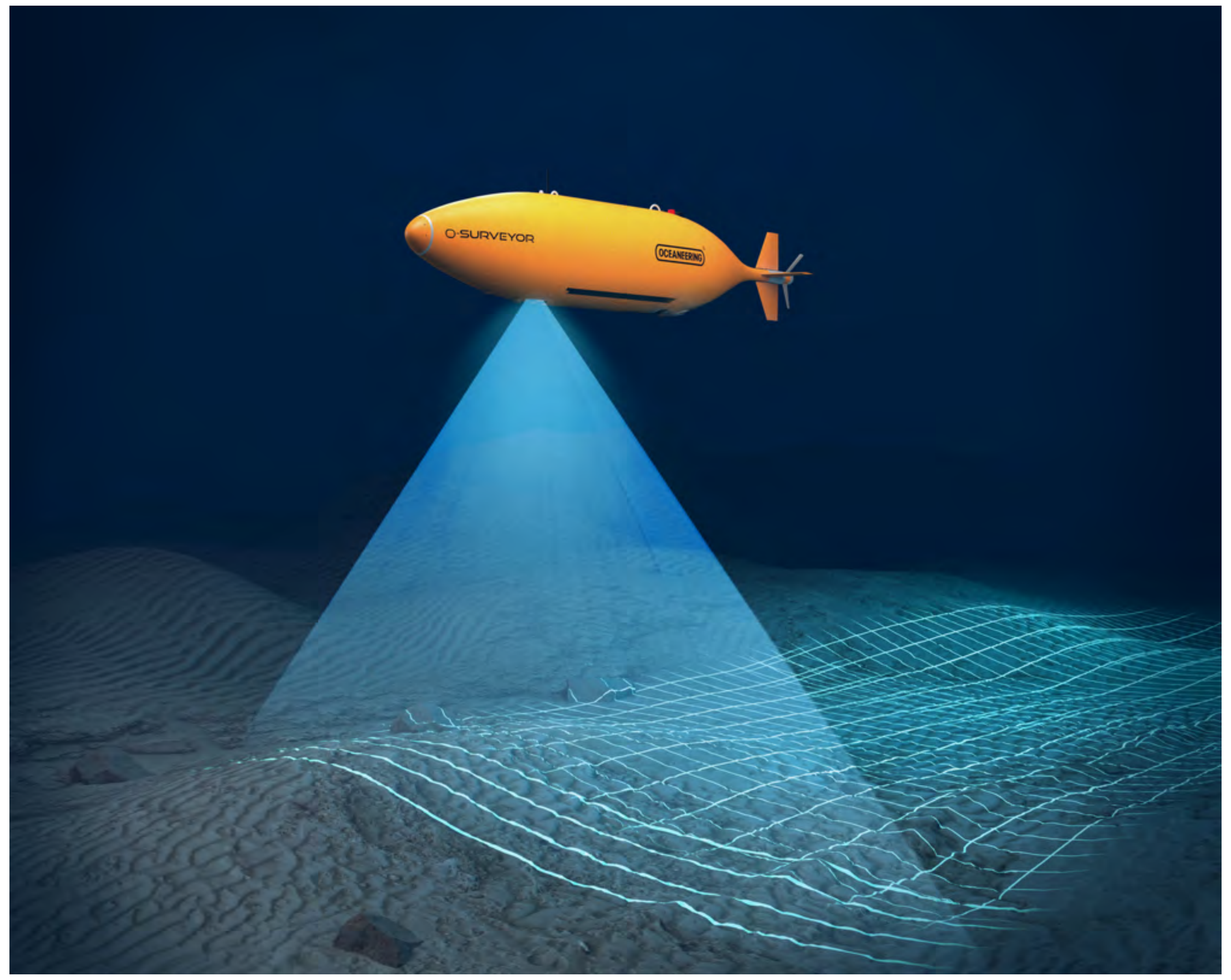
Article | Open Access | Published: 08 December 2021

Learning efficient navigation in vortical flow fields

Peter Gunnarson, Ioannis Mandralis, Guido Novati, Petros Koumoutsakos & John O. Dabiri

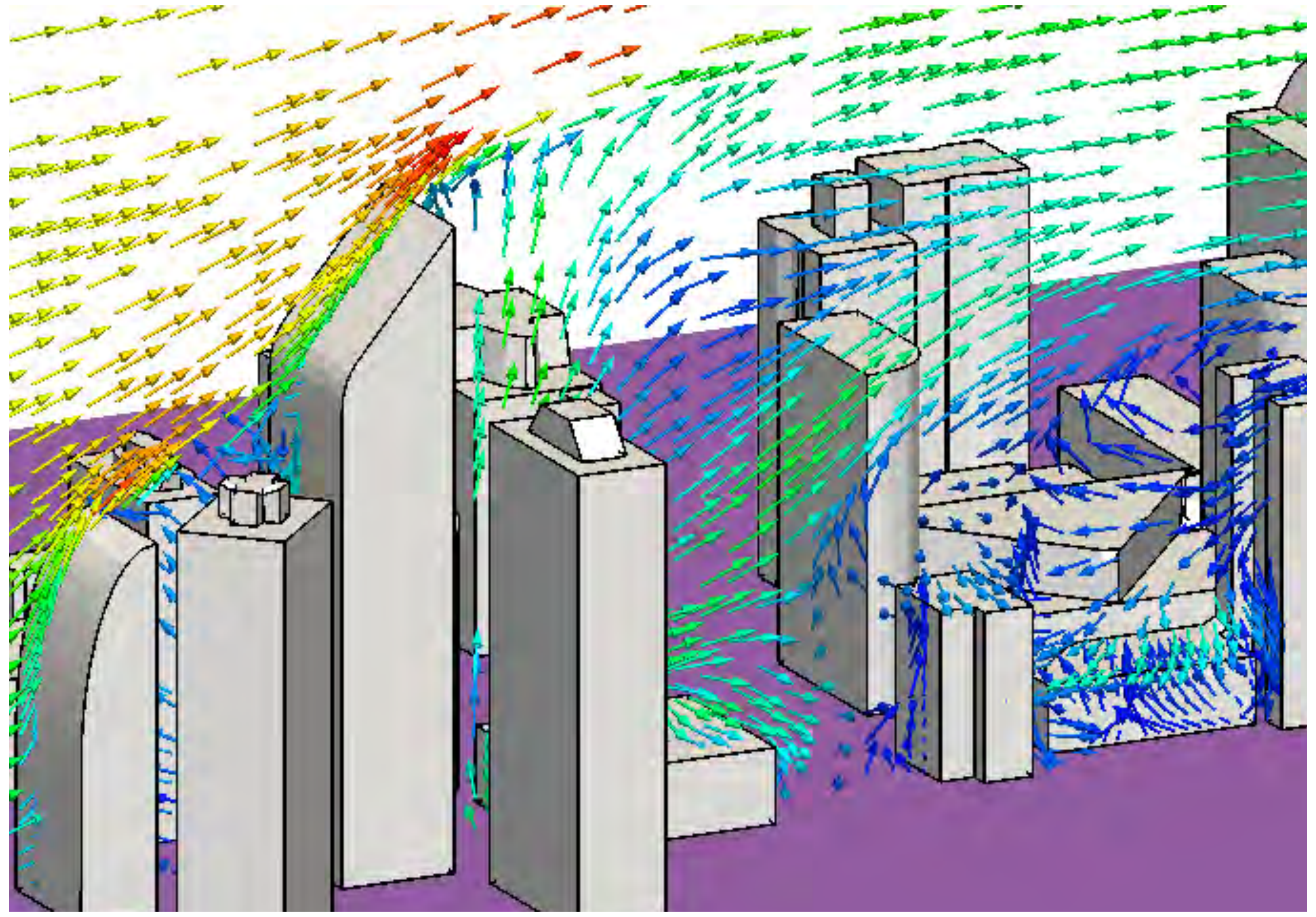
Nature Communications **12**, Article number: 7143 (2021) | Cite this article

In applications, background flow can be:



Unknown

oceaneering.com/survey-and-mapping/



Infeasible to Measure

OpenFoam: windAroundBuildings

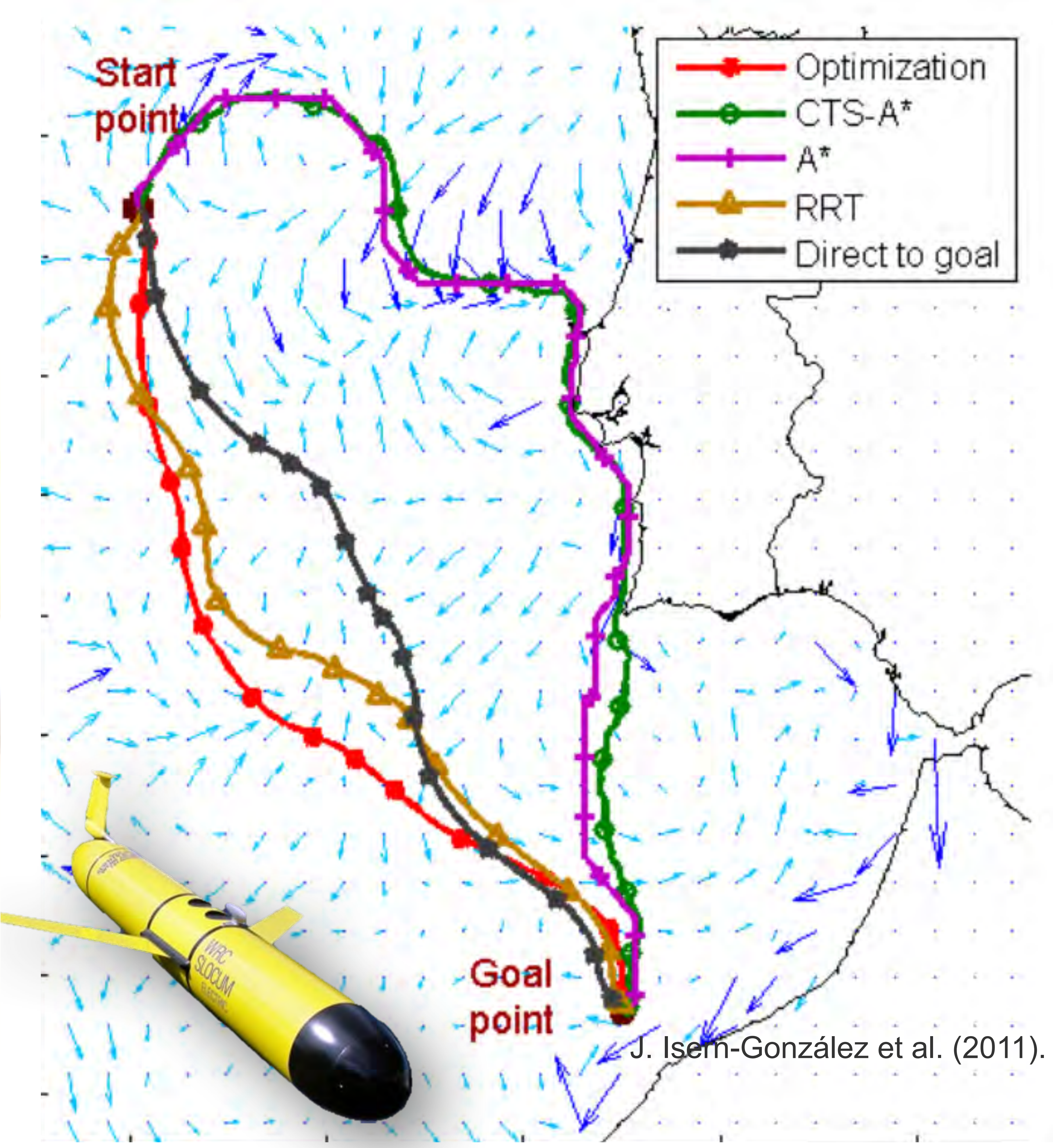
Time-Varying

youtube.com/watch?v=c7mN4xtUssQ

If <u>global flow</u> field is unknown…

… can we still navigate efficiently with <u>limited information</u>?

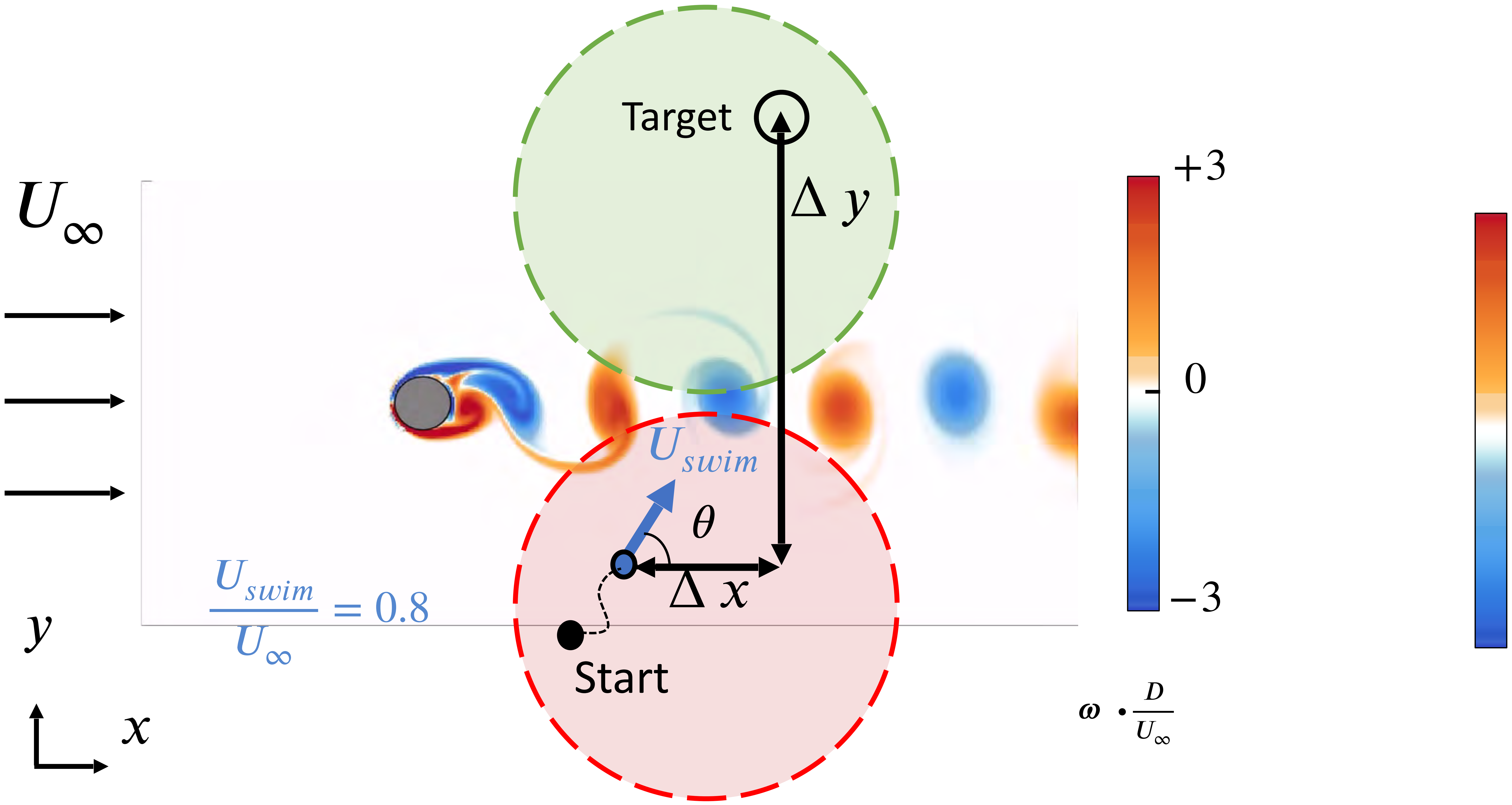
Learning to Navigate through Flow Fields



- with Dabiri group (Caltech)

Problem Description: Navigate Across Wake

$Re = 400$

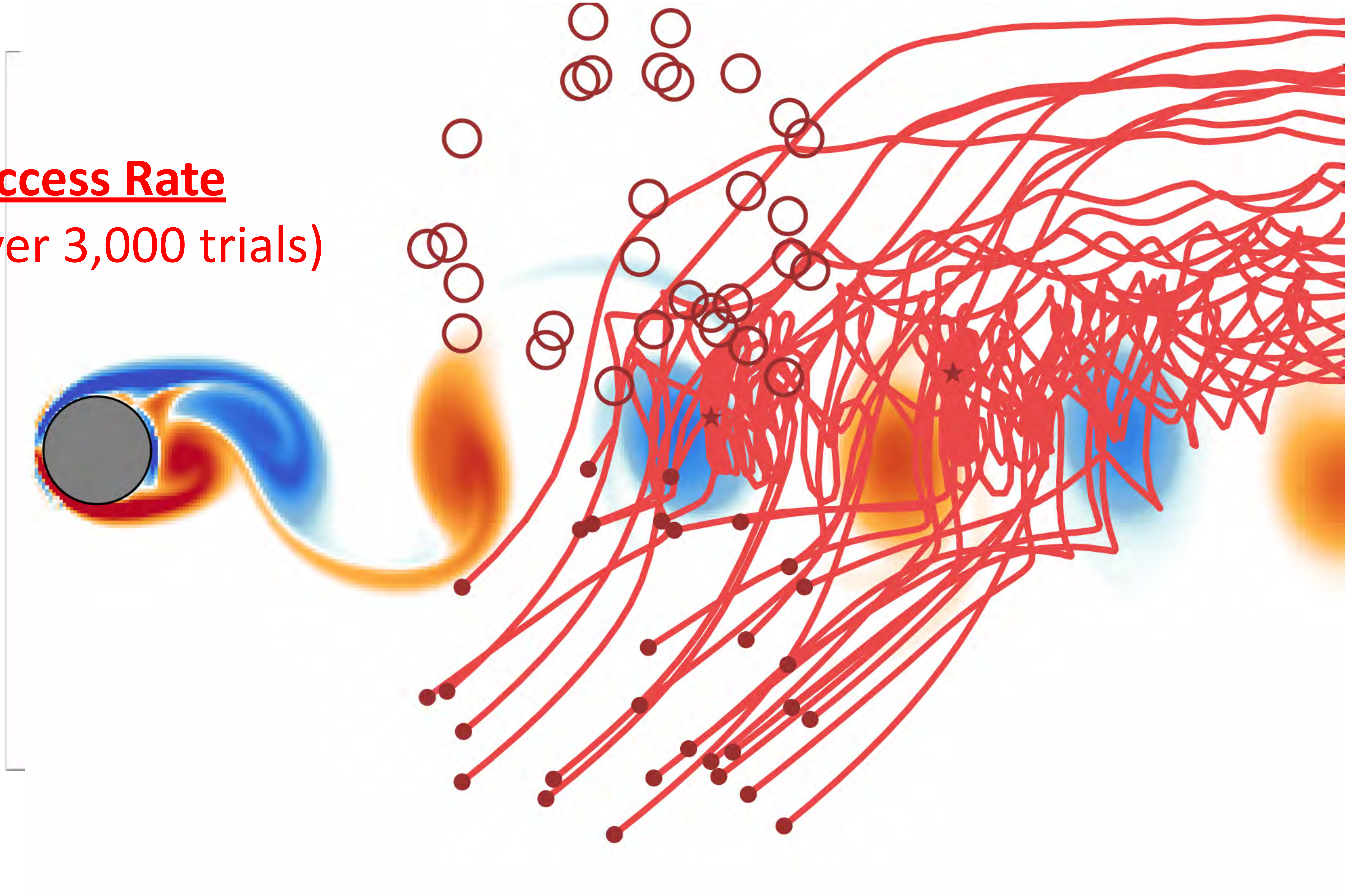


$$\dot{X} = U_s * [\cos\theta, \sin\theta] + U(x, y, t)$$

Swimming towards the target is ineffective

<u>**2% Success Rate**</u>

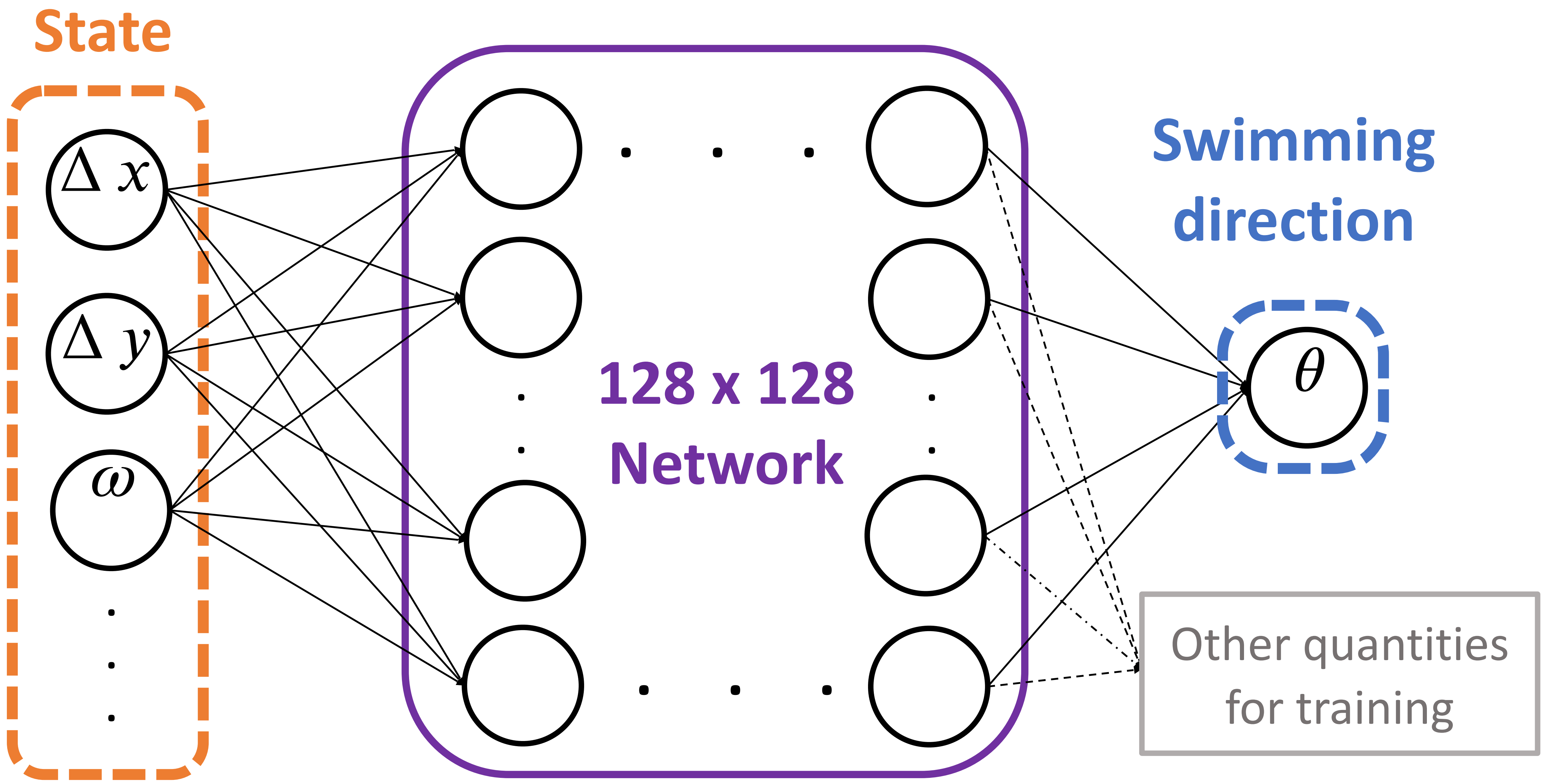
(average over 3,000 trials)

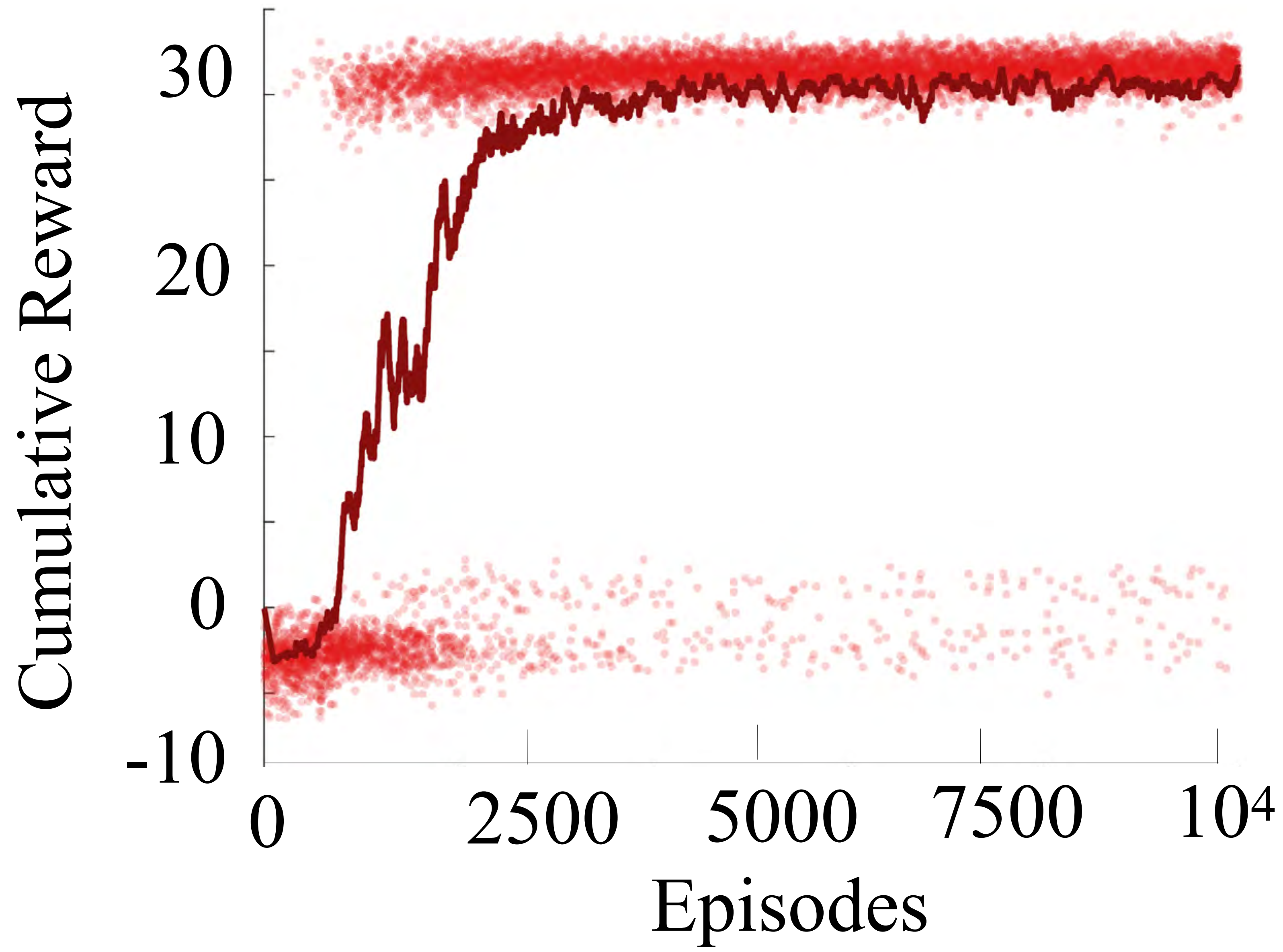


Deep Reinforcement Learning Approach

- Trained using V-RACER algorithm

G. Novati and P. Koumoutsakos (2019)
github.com/cselab/smarties

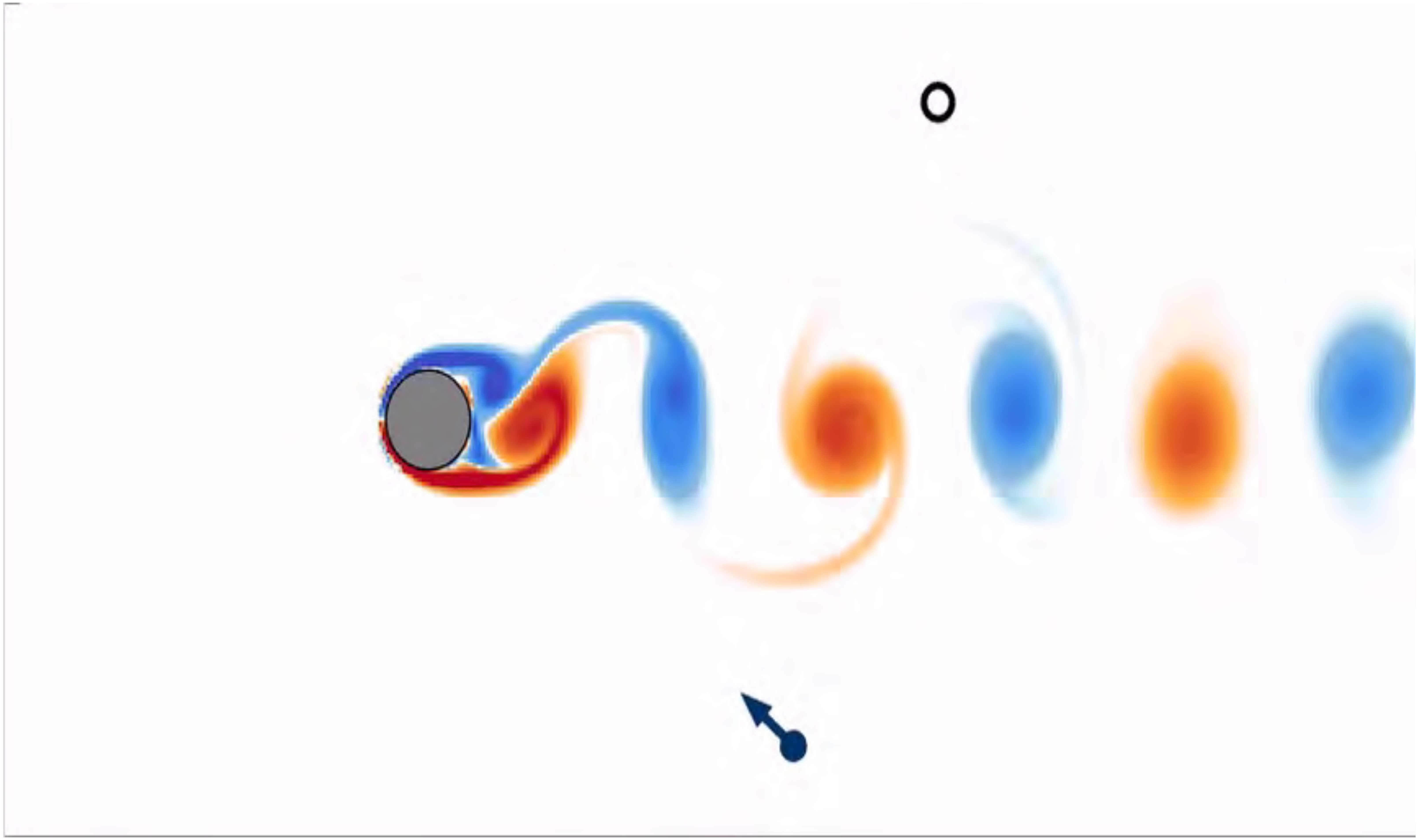




Reward Function: $-U_{swim} \cdot dt - ||(\Delta x_t, \Delta y_t)|| + ||(\Delta x_{t-dt}, \Delta y_{t-dt})||$

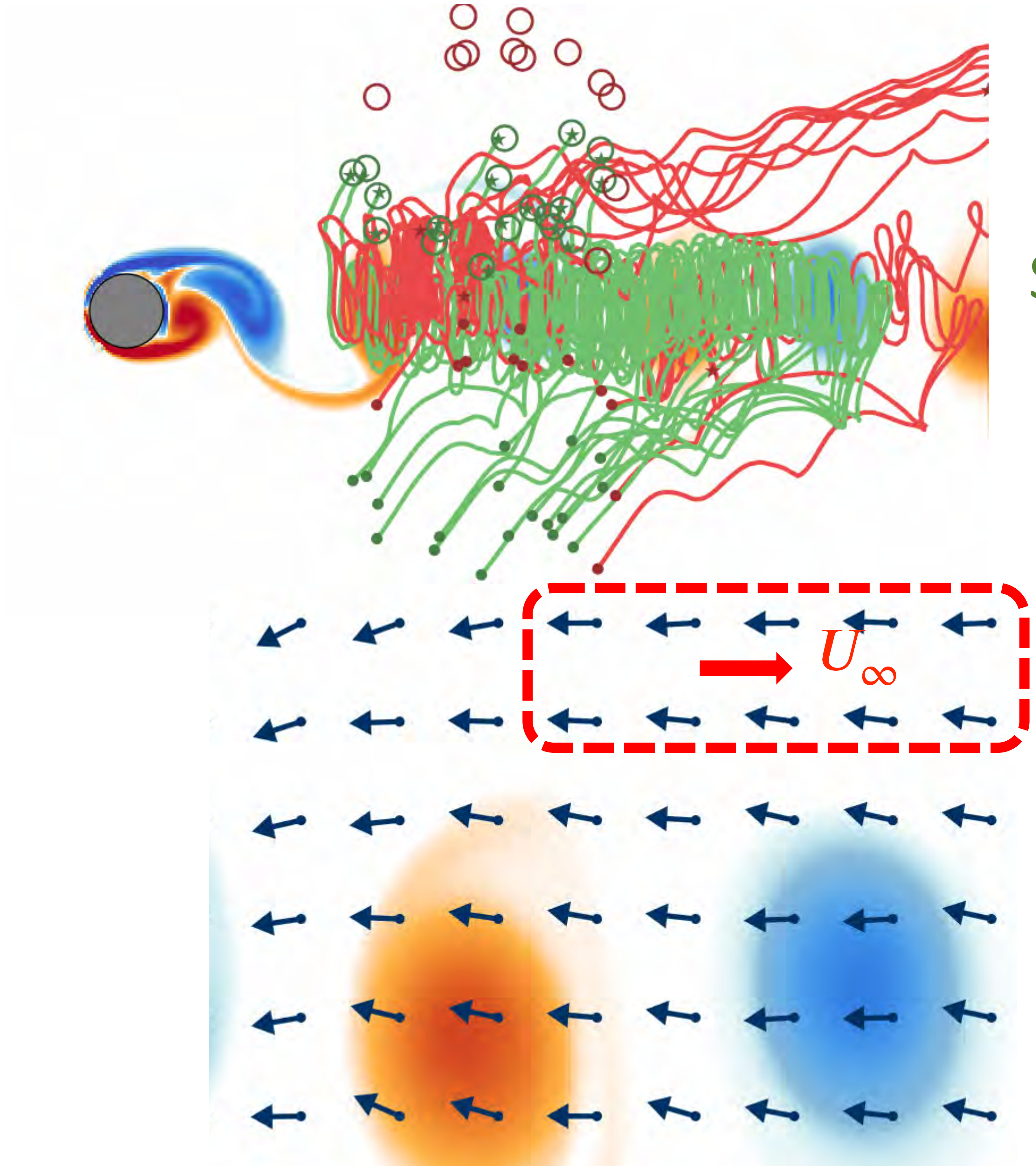
$+30$ (if target reached)

RL Example: $s = \{\Delta x, \Delta y, \omega_t, \omega_{t-dt}\}$



What local flow information is helpful?

Position + Local Vorticity: $\{\Delta x, \Delta y, \omega_t, \omega_{t-dt}\}$

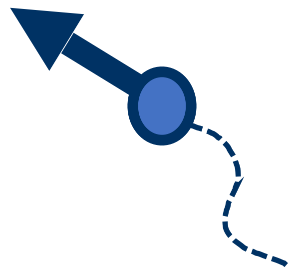


53% Success Rate

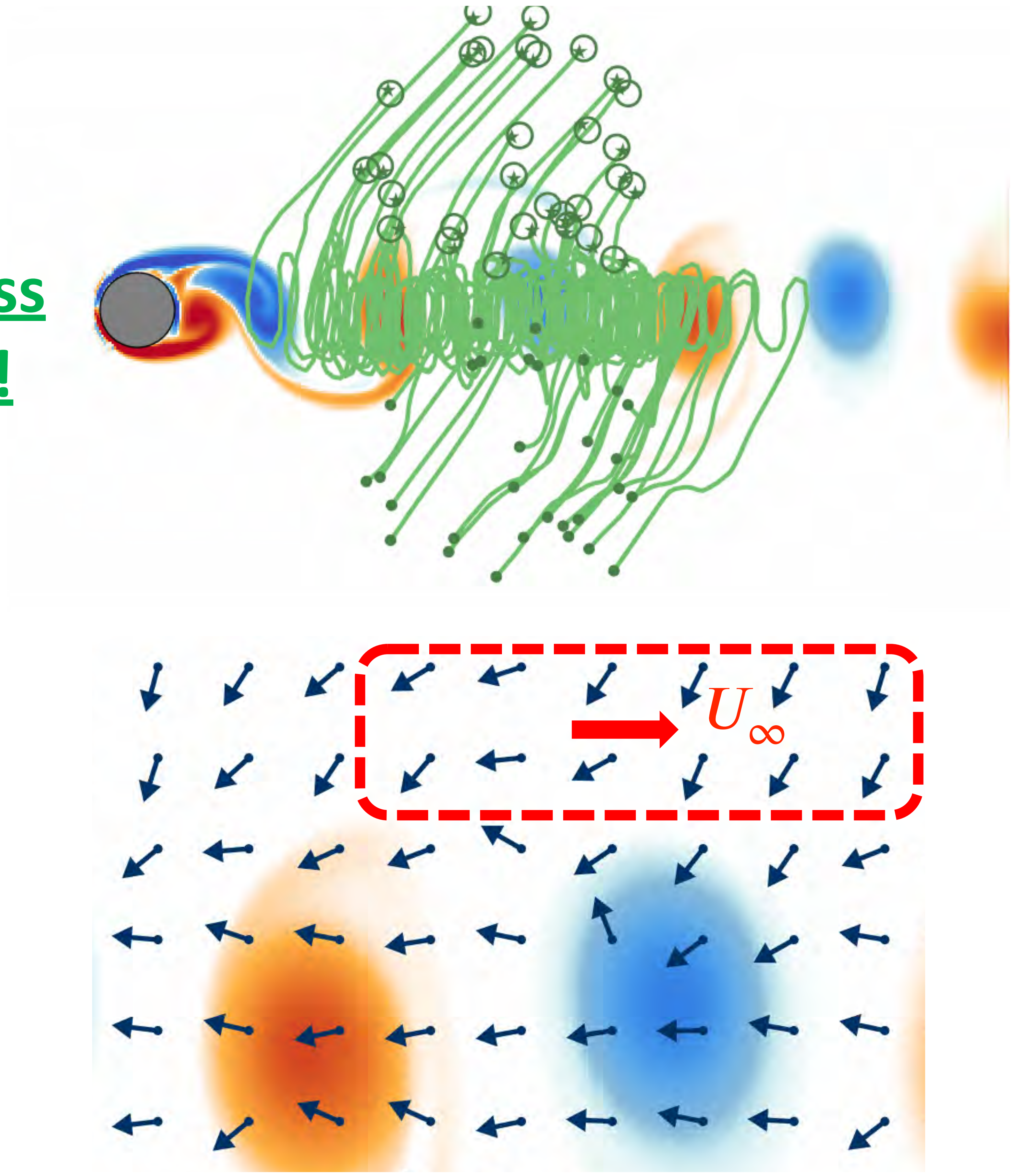
Moderate response to background flow

Swimming Direction

$U_{swim} = 0.8U_\infty$



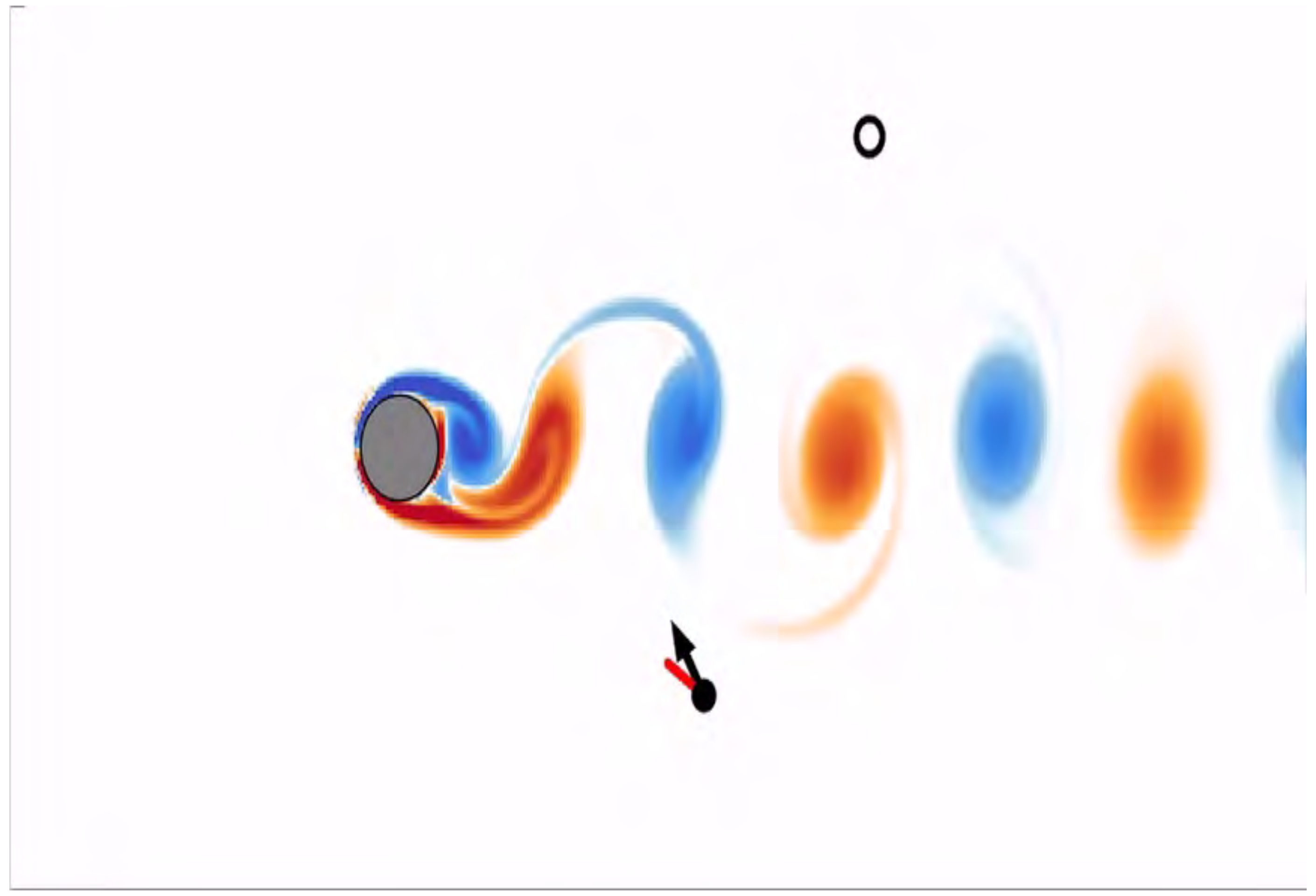
Position + Local Velocity: $\{\Delta x, \Delta y, u, v\}$



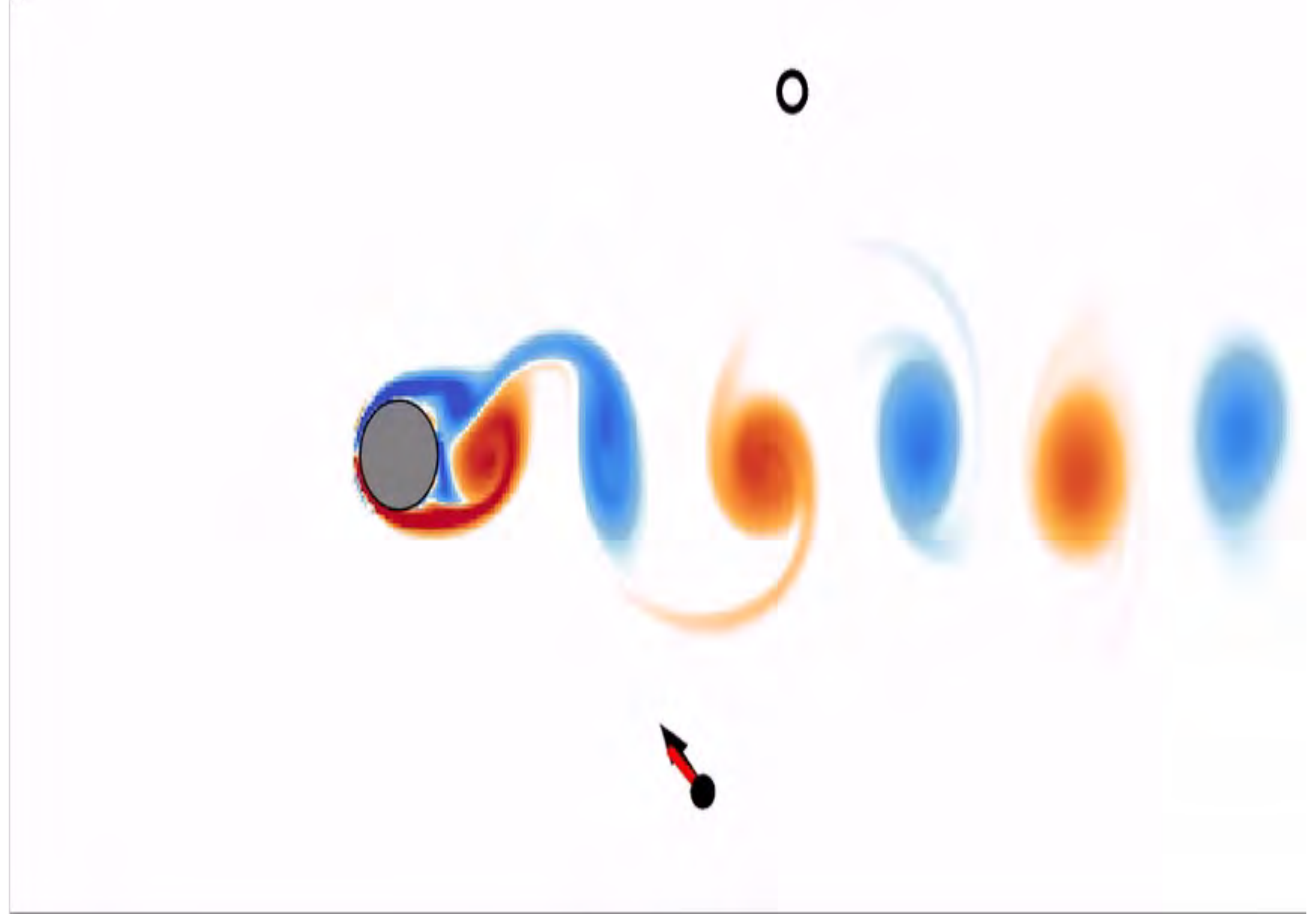
99% Success Rate!

Increased sensitivity to background flow

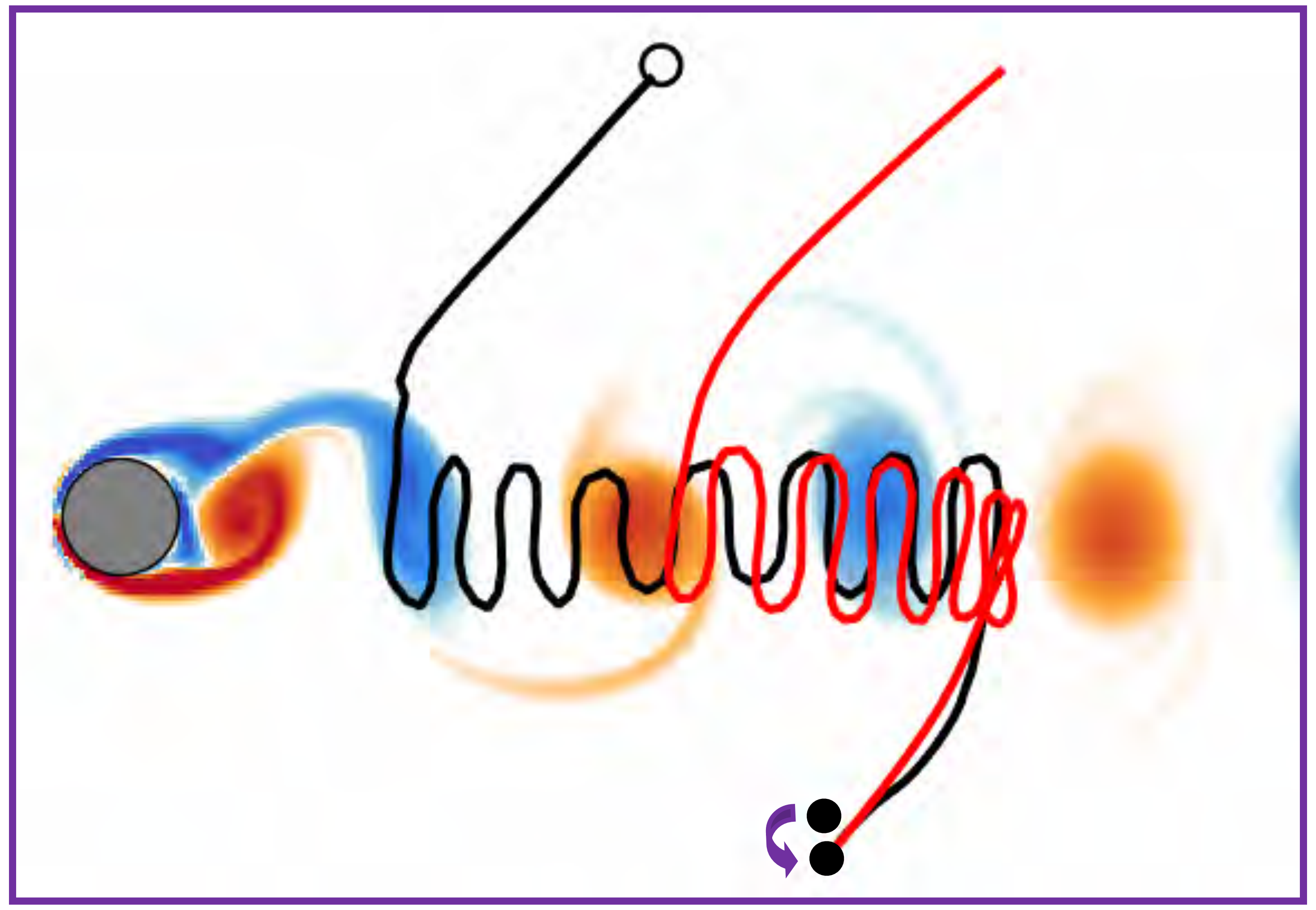
Comparison with Optimal Control* for three example start/goal points



RL: T_f = 10.4
Optimal: T_f = 9.22
(11% faster)



RL: T_f = 23.47
Optimal: T_f = 17.78
(24% faster)



RL: T_f = 33.6
Optimal: T_f = 31.07
(8% faster)

<u>Optimal Control is faster, but:</u>

- Needs global flow field
- Needs to be recalculated for every new start/target

*Using RRT (Lavalle 1998) + constrained gradient descent (fconmin in MATLAB)

FISH SCHOOLING

- Behavioral Traits - Vortex Dynamics
- Energetic benefits ?

- **Why do fish swim in groups?**
  A. Propulsive advantage?
  B. Social behavior?
  C. ……

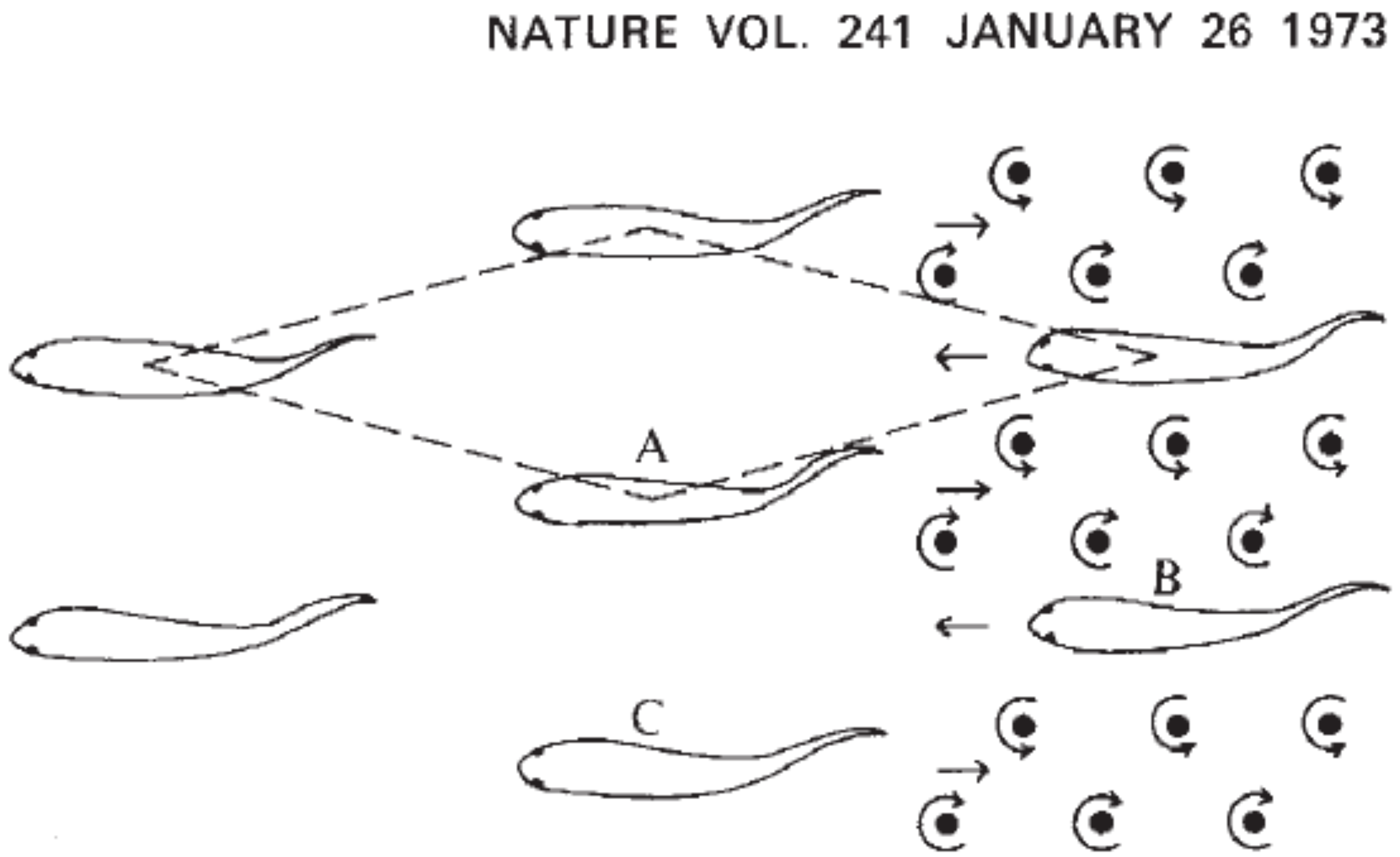
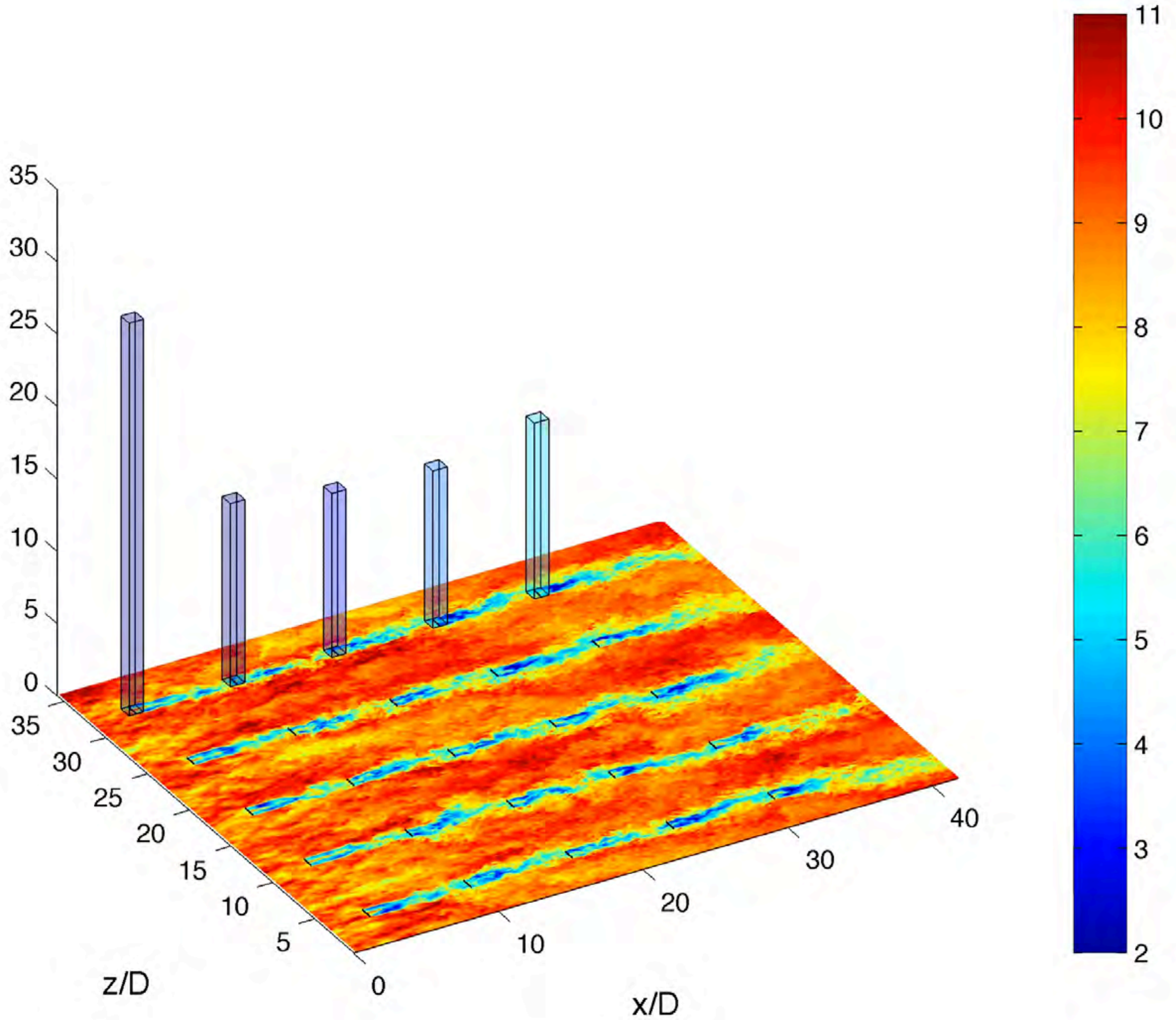
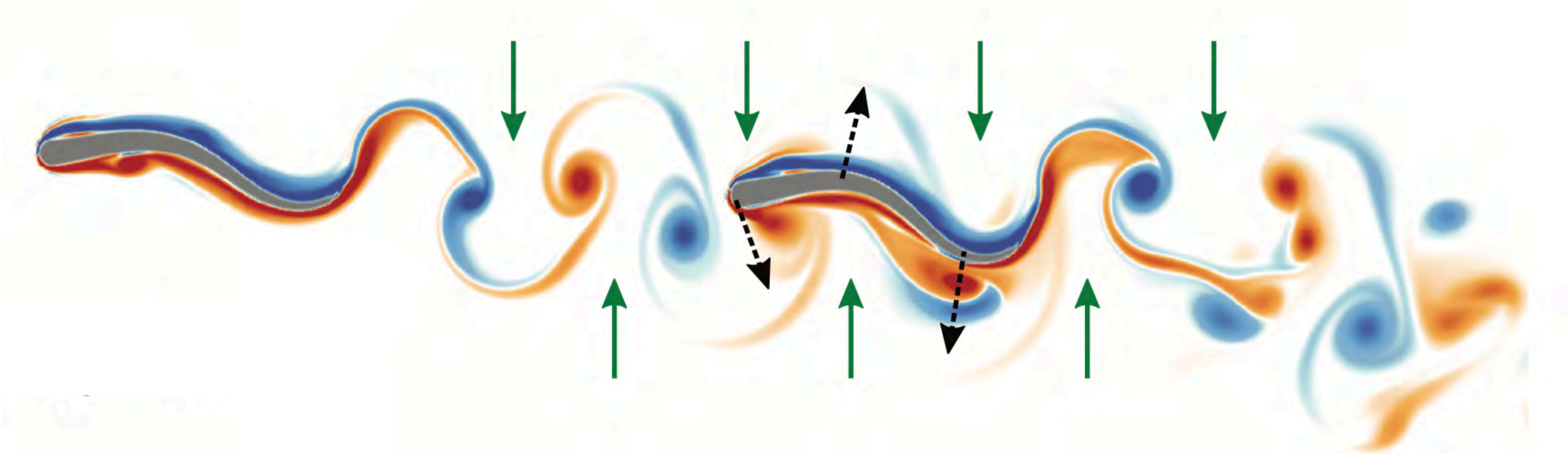


Fig. 1 Part of a horizontal layer of fish in a school, from above. Arrows near vortex streets show direction of induced flow relative to the vortices. The dotted line shows a "diamond" pattern.

- **What can WE learn ?**
  - Hydrodynamics vs Behaviour ?
  - Engineering based on Collective Hydrodynamics ?
  - ….

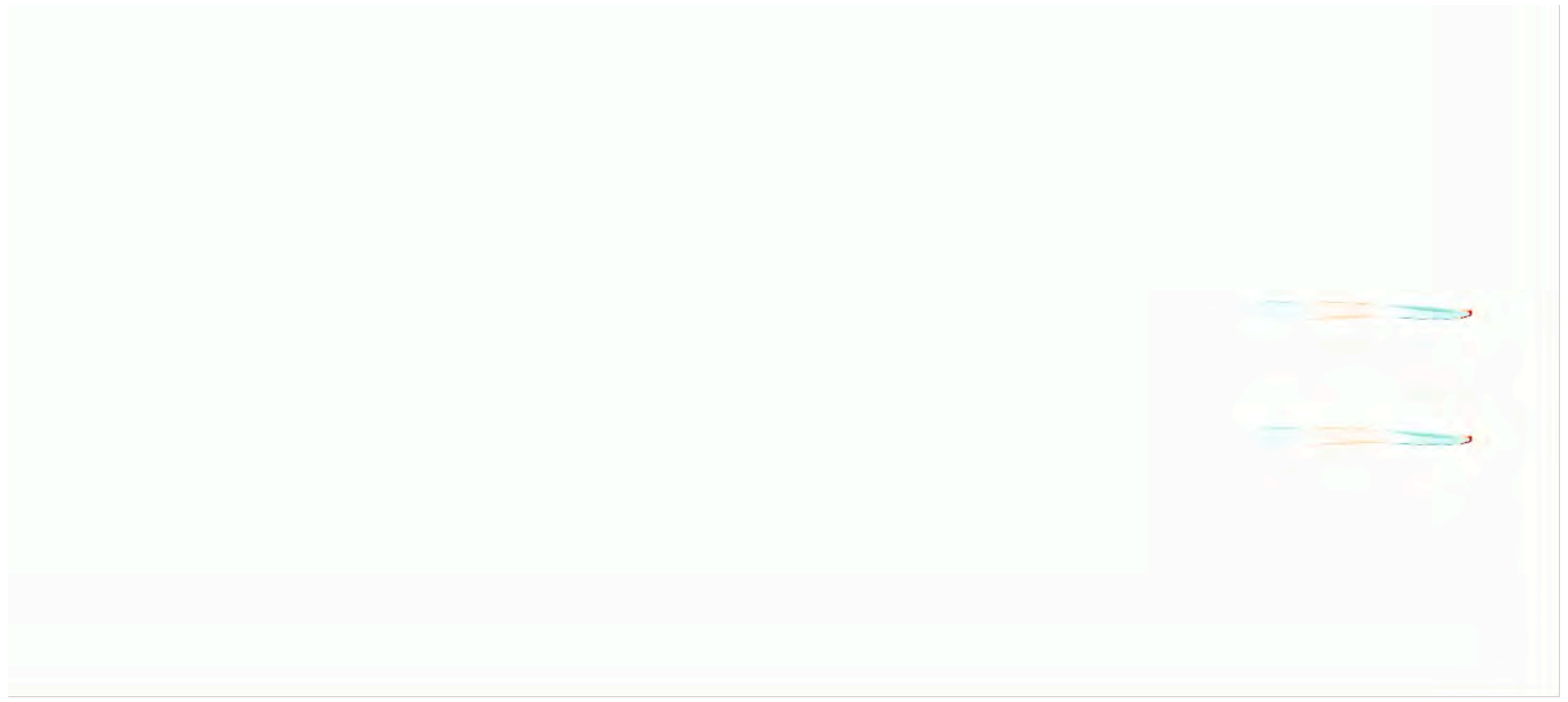


Two FISH



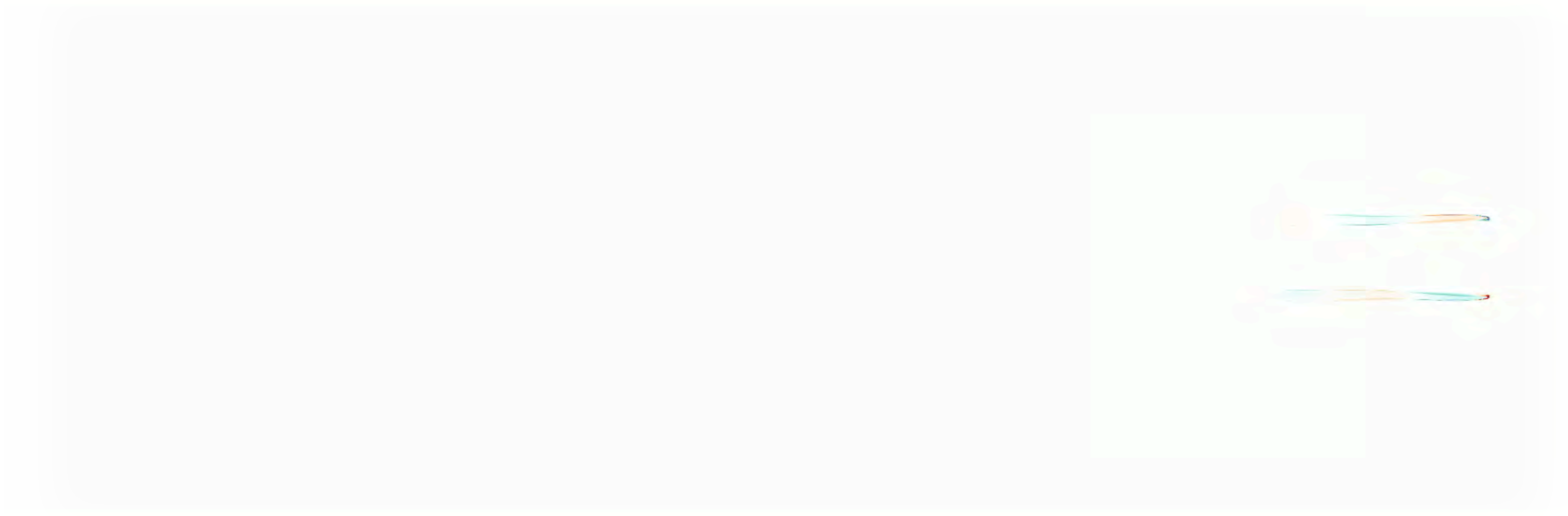
The Challenge of Schooling Fish

in phase fish

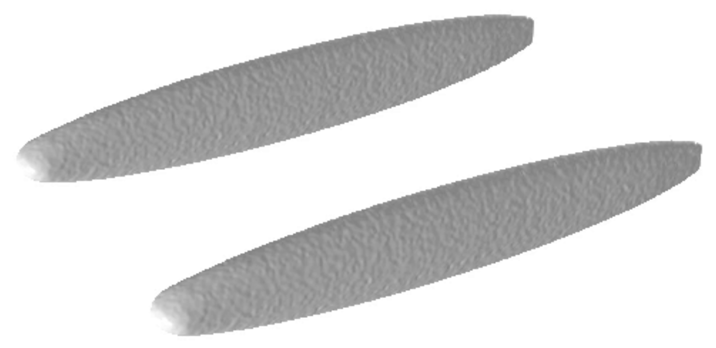


aligned school

Fixed motion pattern : Reverse Engineering Fails



out of phase fish (π)(2D)



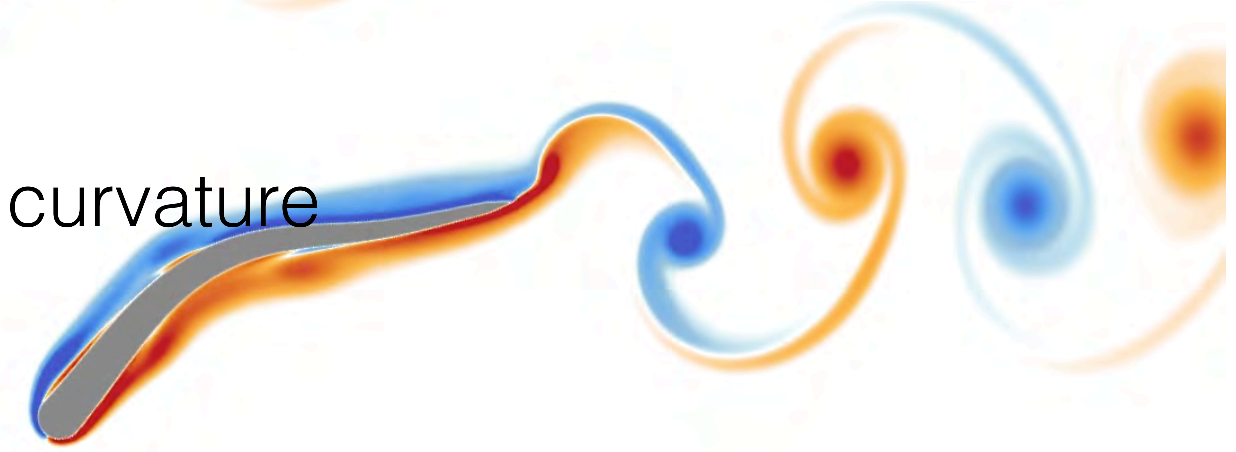
out of phase fish (π)(3D)

Synchronisation through Learning for Self-propelled Swimmers

ACTIONS: **Modulate** body deformation

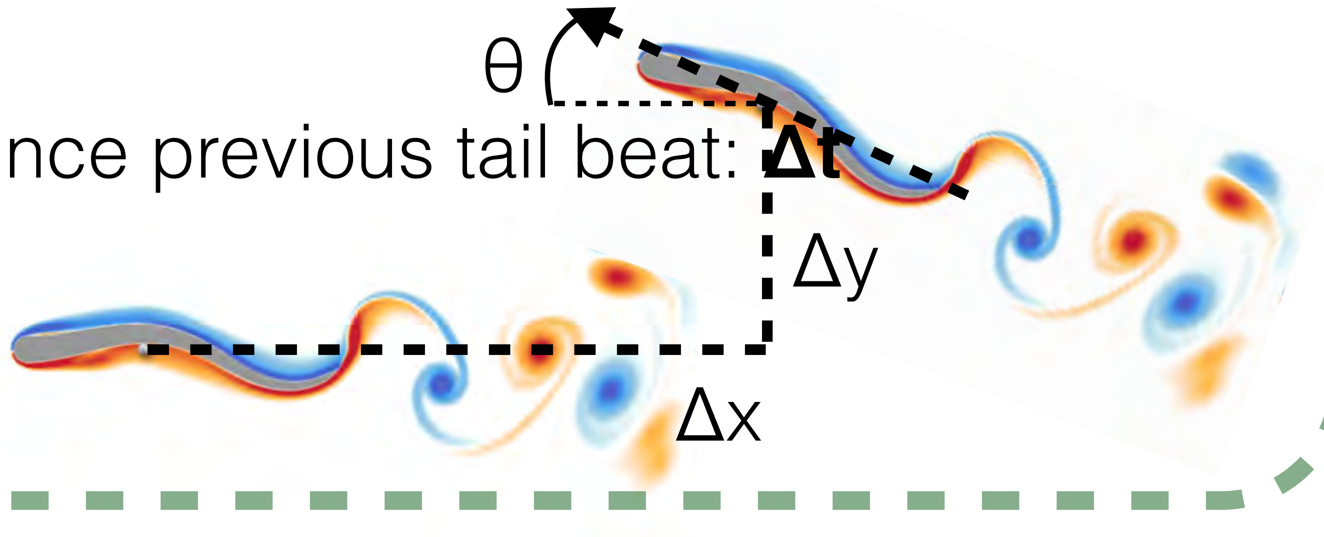
- Increase curvature

- Decrease curvature

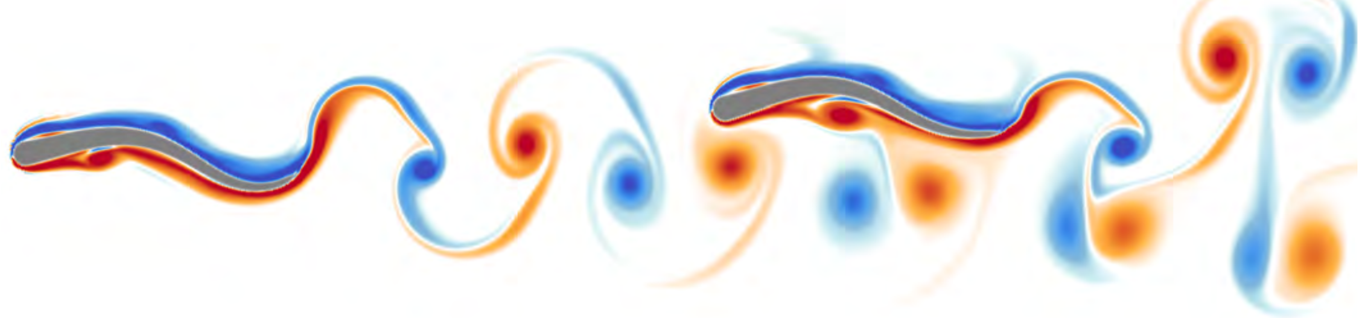


STATES

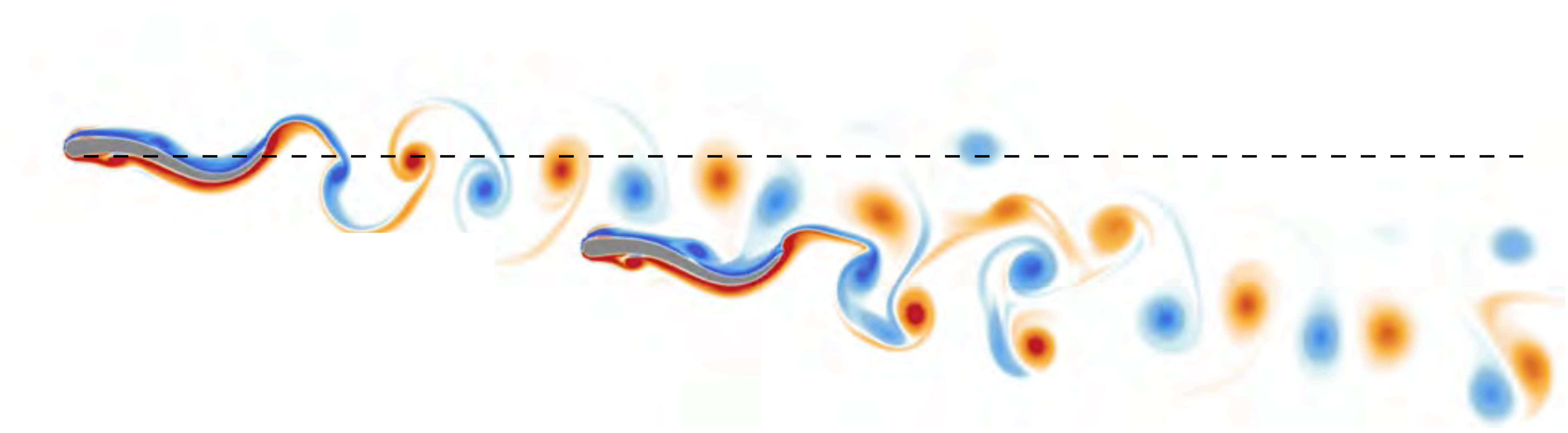
- Orientation relative to leader: **Δx, Δy, θ**
- Time since previous tail beat: **Δt**
- Current manoeuvre



REWARD :distance to leader OR swimming-efficiency

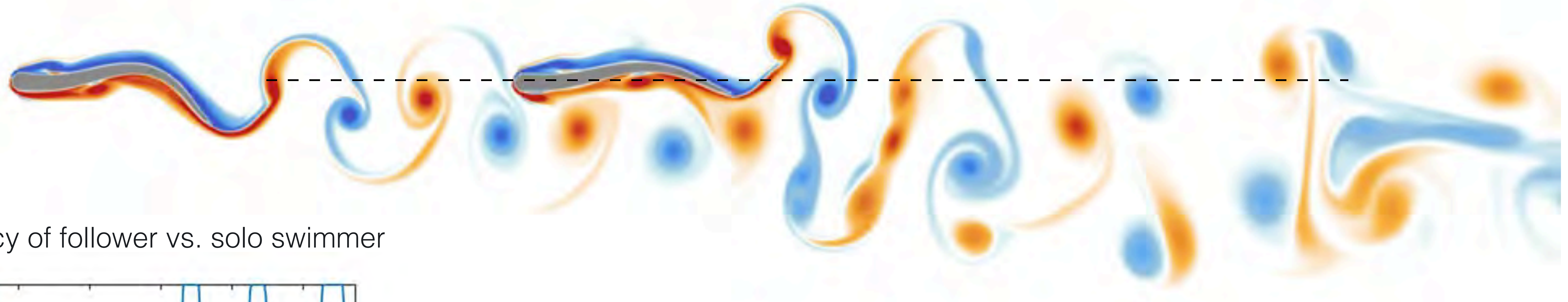


GOAL I : minimize Δy

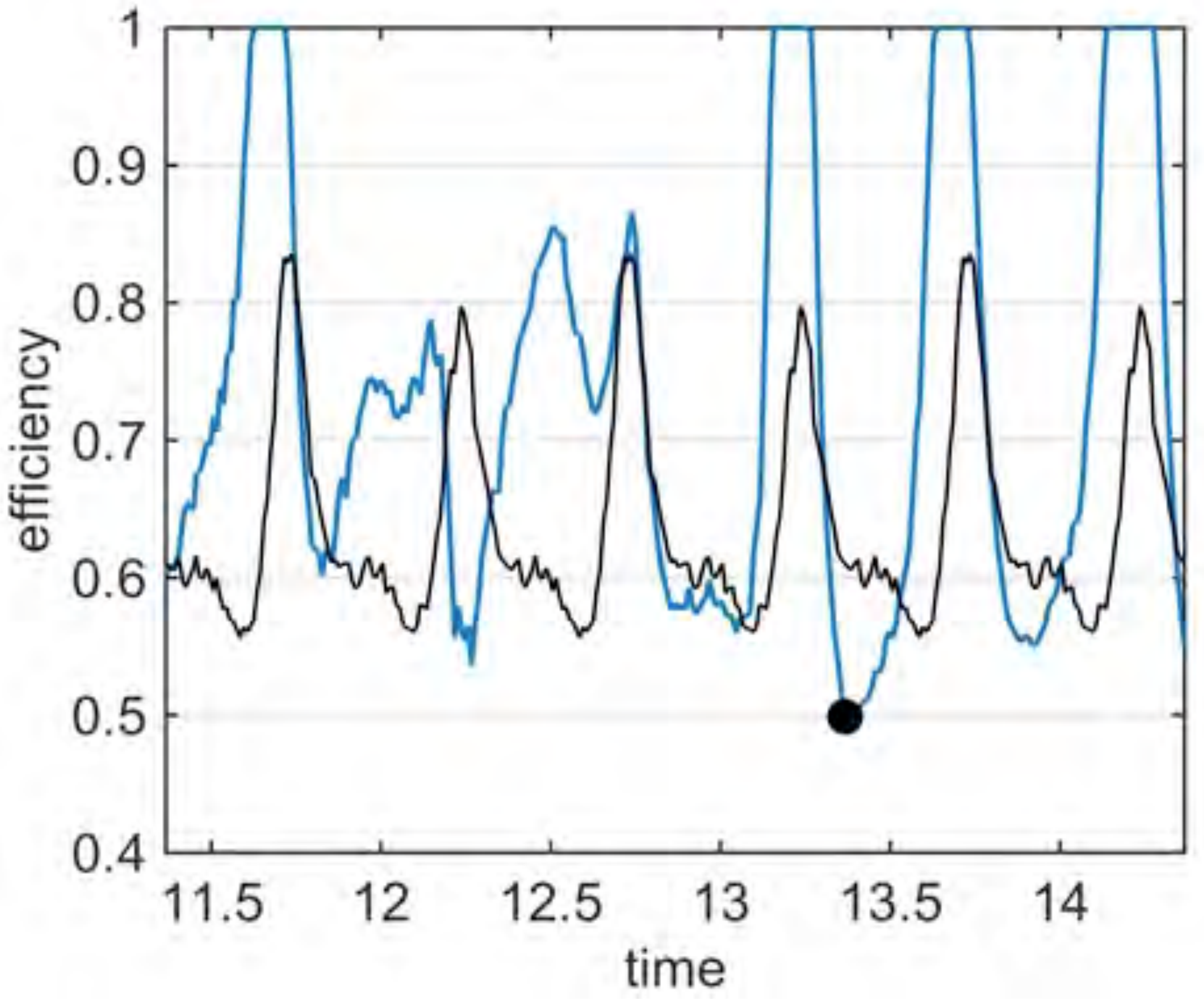


- *Smart Follower:*
  - *Stay in the leader's wake*
  - *Steer and course correct*
  - ***P_{def} drops** : decreased fluid resistance due to assistance from vortices*

IS IT EFFICIENT TO SWIM IN A WAKE ?



Efficiency of follower vs. solo swimmer



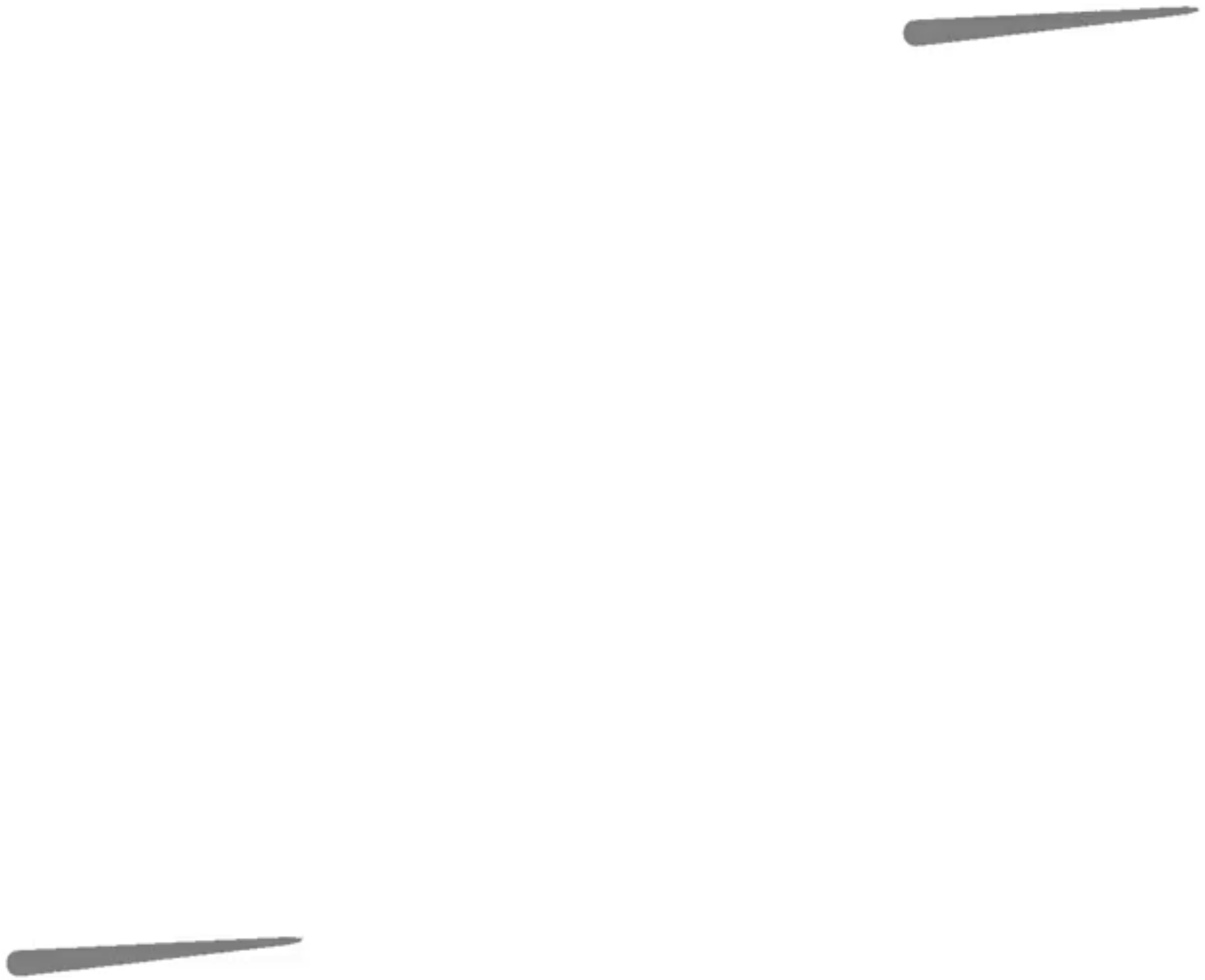
$$\text{efficiency} = \frac{Tu}{Tu + max(\int \mathbf{F} \cdot \mathbf{u}_{def} dS, 0)}$$

- Average **efficiency increased by 11.8**%
- Increased efficiency was not a learning objective:
  - Emerges from learning to stay in leader's wake

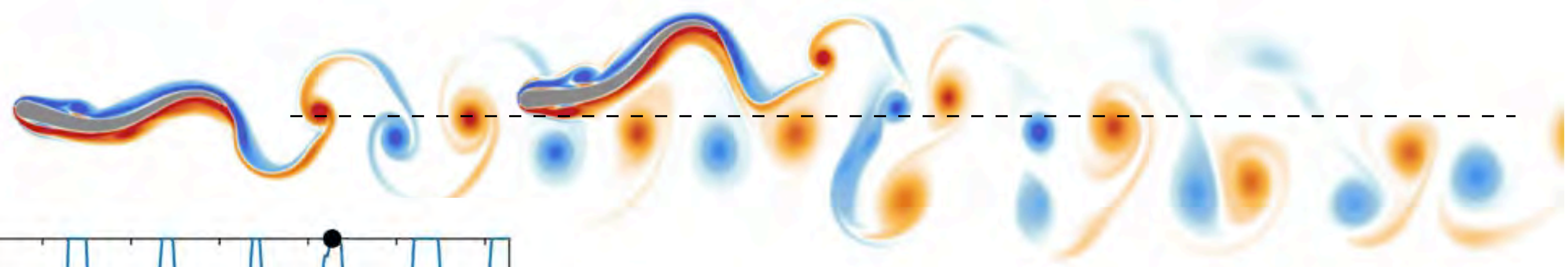
GOAL II : maximising efficiency

No distance-based constraints specified

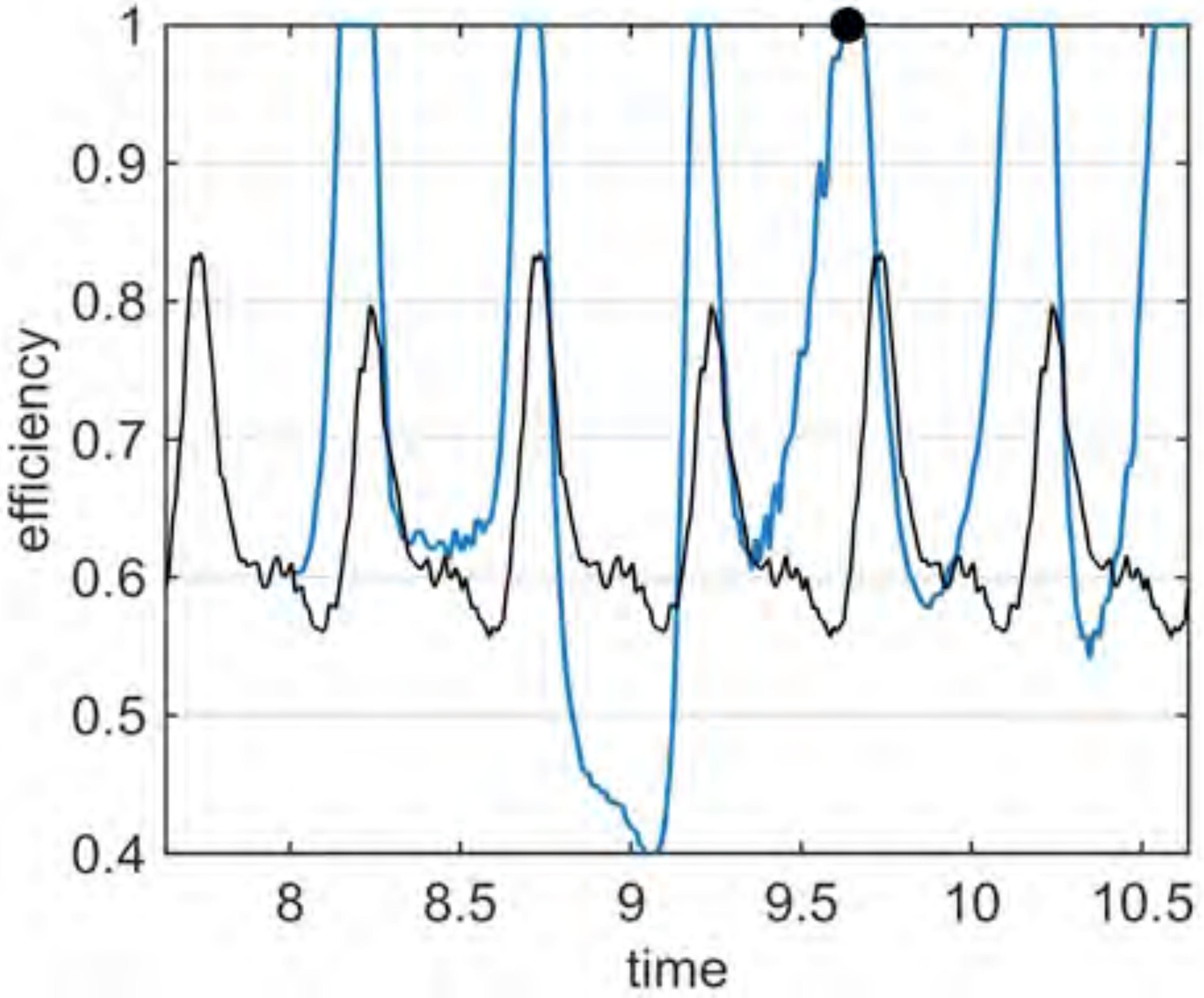
EARLY STAGES OF LEARNING



GOAL II : MAX EFFICIENCY



No distance constraints specified

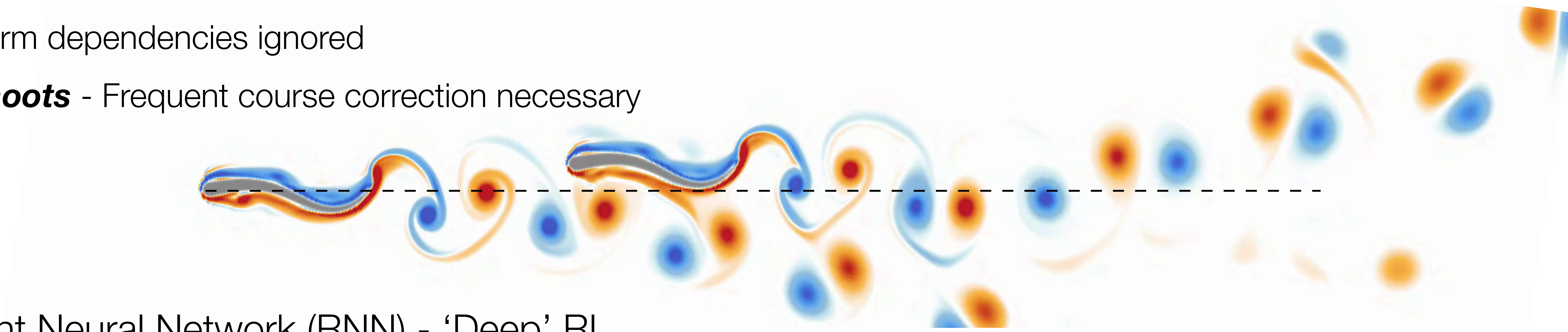


$$R_\eta = \frac{Tu}{Tu + max(P_{def}, 0)}$$

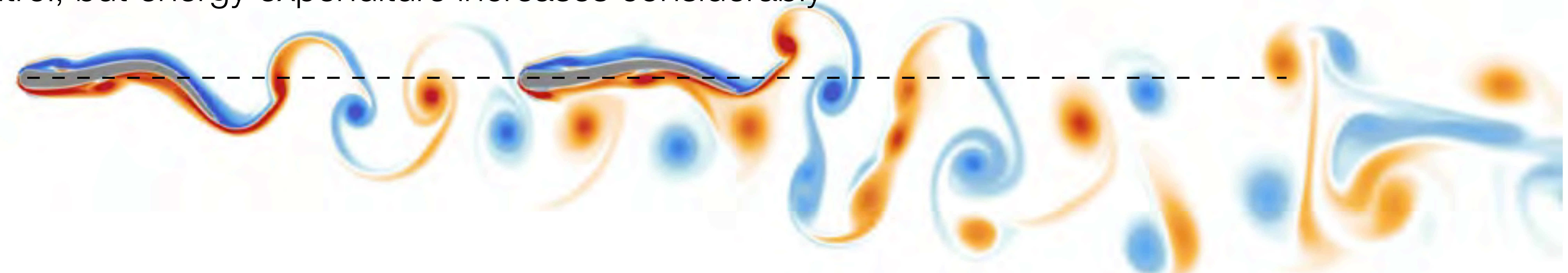
- Follower ***<u>opts</u>*** to interact with wake-vortices
  - *Overall, 28% gain in average efficiency*

Influence of Time-history

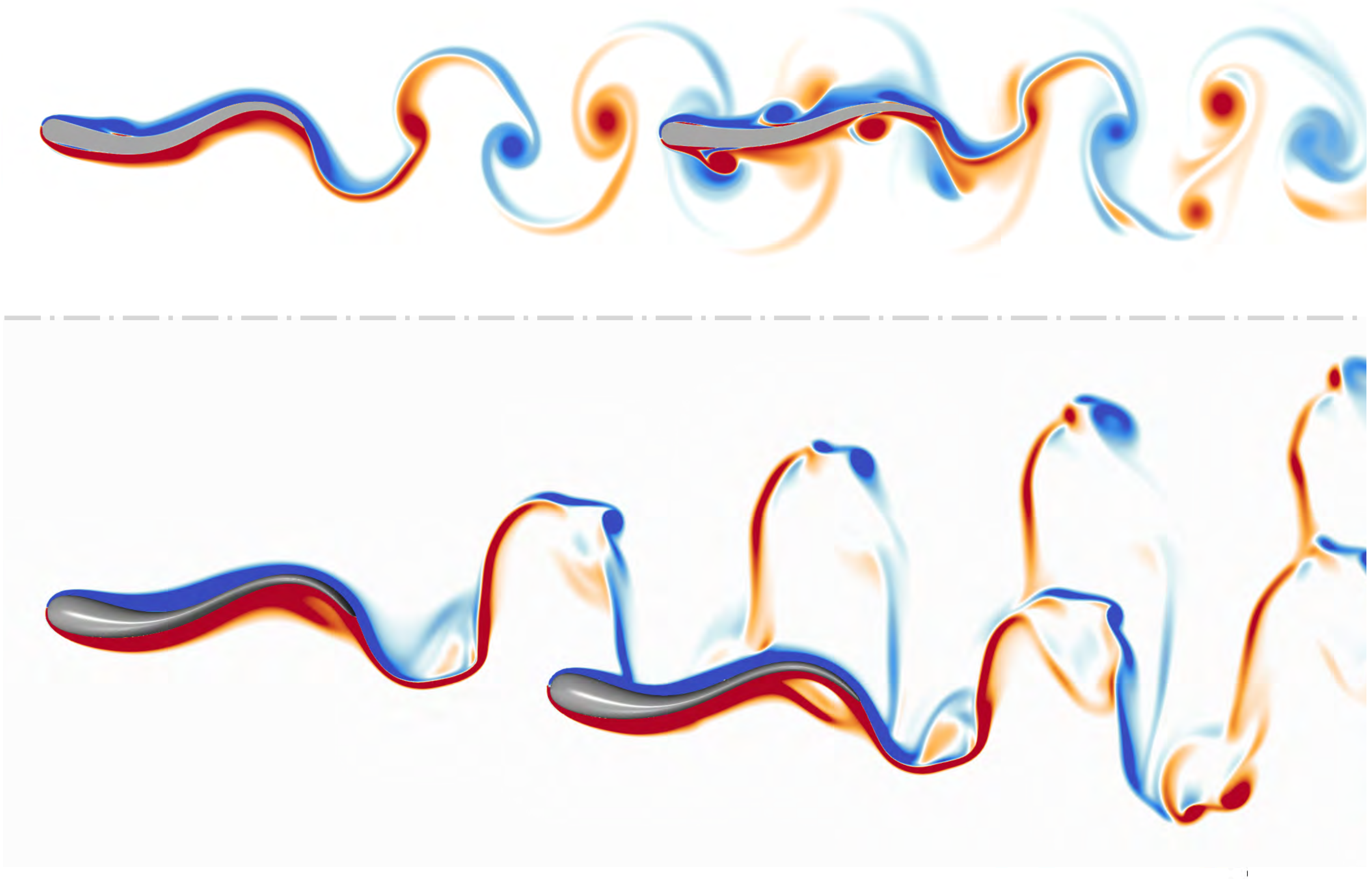
- Feed Forward Network (FFN) - 'Vanilla' RL
  - Action depends *explicitly* on the current state only
  - Long-term dependencies ignored
  - ***Overshoots*** - Frequent course correction necessary



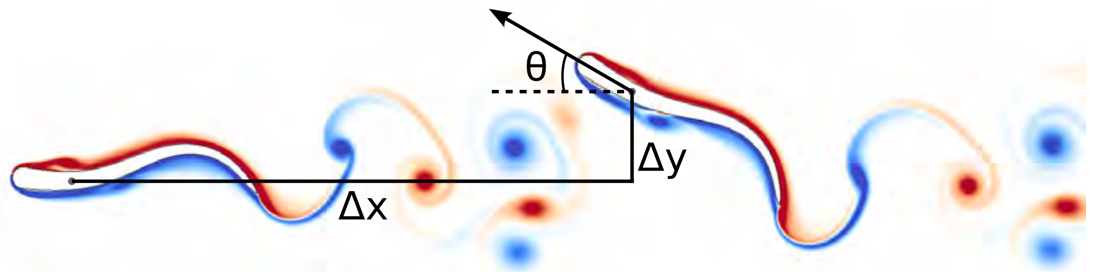
- Recurrent Neural Network (RNN) - 'Deep' RL
  - Action depends on information from agent's time-history
  - Agent ***anticipates*** interaction with the flow - more robust control
  - Better attitude-control, but energy expenditure increases considerably



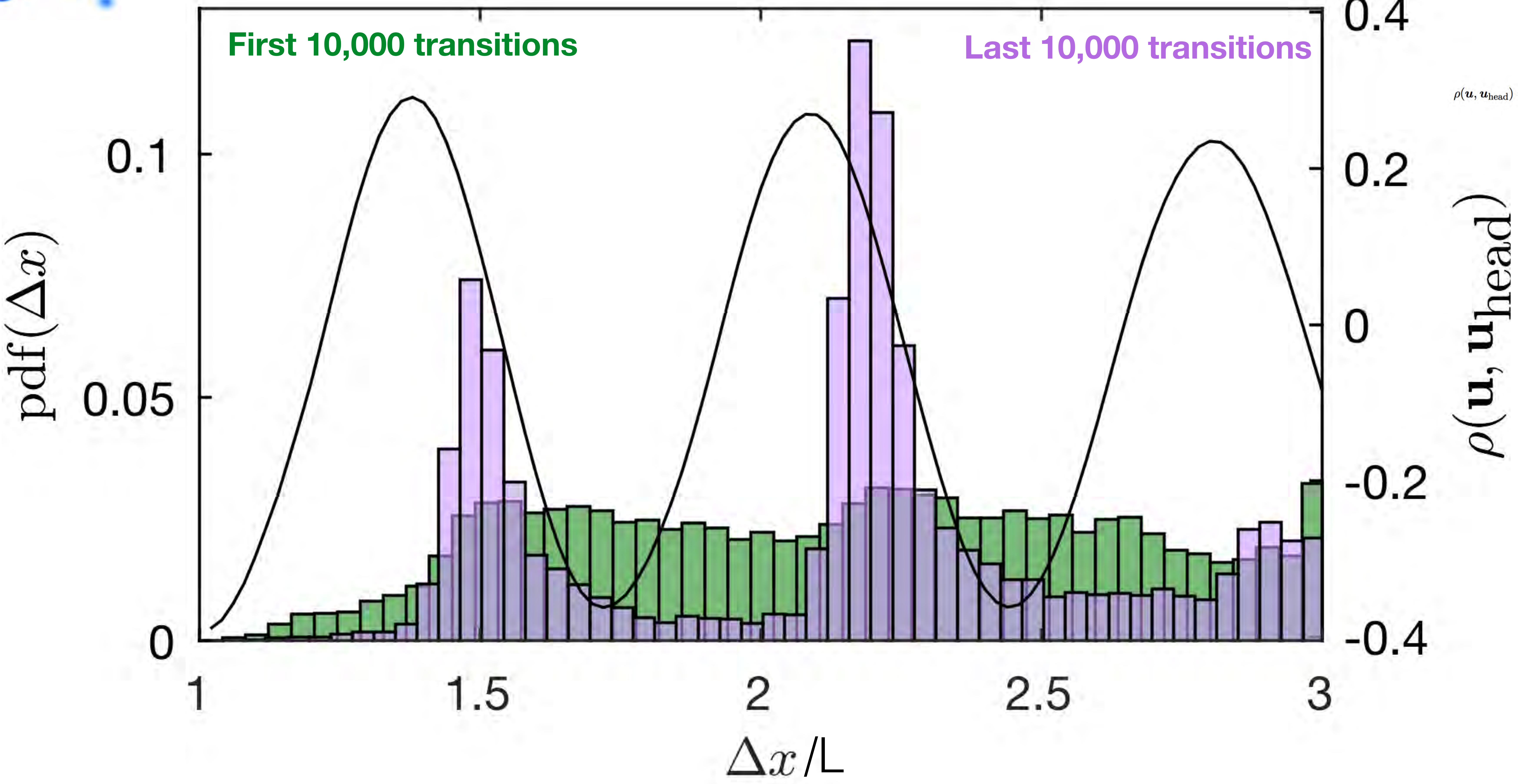
3D Schools



What can **WE** learn from 'smart' swimmers?



How does behaviour evolve ***during*** training?

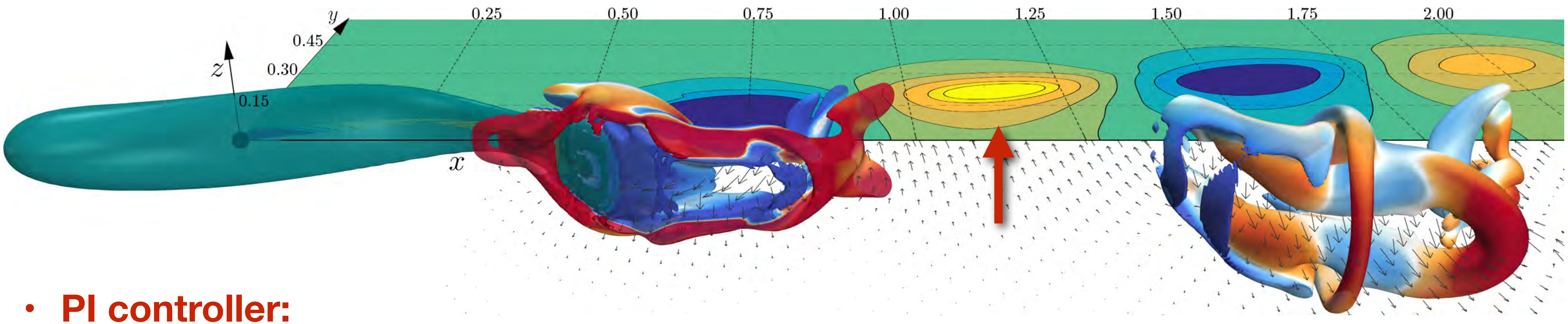


$$\rho(\boldsymbol{u}, \boldsymbol{u}_{\text{head}}) = \frac{\operatorname{cov}(\boldsymbol{u}(x,y), \boldsymbol{u}_{\text{head}})}{\sigma_{\boldsymbol{u}(x,y)}\ \sigma_{\boldsymbol{u}_{\text{head}}}} = \frac{\sum_t \boldsymbol{u}(x,y,t) \cdot \boldsymbol{u}_{\text{head}}(t)}{\sqrt{\sum_t \|\boldsymbol{u}(x,y,t)\|^2\ \sum_t \|\boldsymbol{u}_{\text{head}}(t)\|^2}}$$

$$\rho(\boldsymbol{u}, \boldsymbol{u}_{\text{head}}) = \frac{\operatorname{cov}(\boldsymbol{u}(x,y), \boldsymbol{u}_{\text{head}})}{\sigma_{\boldsymbol{u}(x,y)}\ \sigma_{\boldsymbol{u}_{\text{head}}}} = \frac{\sum_t \boldsymbol{u}(x,y,t) \cdot \boldsymbol{u}_{\text{head}}(t)}{\sqrt{\sum_t \|\boldsymbol{u}(x,y,t)\|^2\ \sum_t \|\boldsymbol{u}_{\text{head}}(t)\|^2}}$$

averaged over one tail beat

$$\rho(\boldsymbol{u}, \boldsymbol{u}_{\text{head}}) = \frac{\operatorname{cov}\left(\boldsymbol{u}(x,y), \boldsymbol{u}_{\text{head}}\right)}{\sigma_{\boldsymbol{u}(x,y)}\ \sigma_{\boldsymbol{u}_{\text{head}}}} = \frac{\sum_t \boldsymbol{u}(x,y,t)\cdot \boldsymbol{u}_{\text{head}}(t)}{\sqrt{\sum_t \|\boldsymbol{u}(x,y,t)\|^2\ \sum_t \|\boldsymbol{u}_{\text{head}}(t)\|^2}}$$



- **PI controller:**
  - Modulate follower's undulations (curvature + amplitude)
  - Maintain the target position specified

Follower: ~20% more efficient

Verma et al., PNAS, 2018

Verma et al., PNAS, 2018

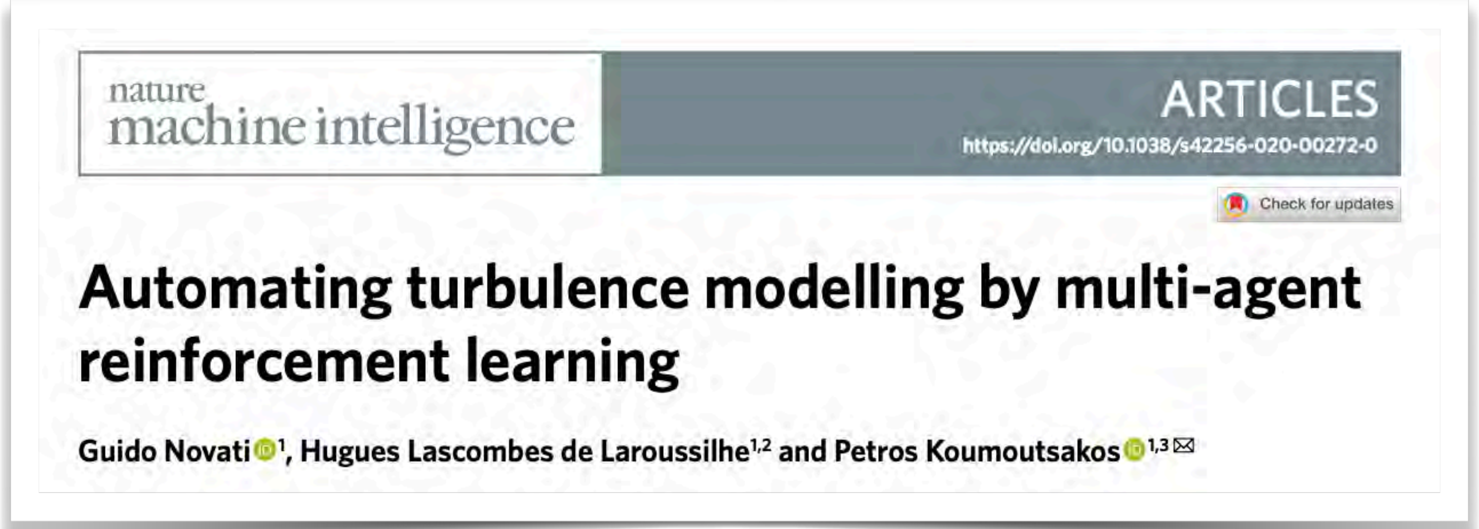
ABF catching a circulating cancerous cell

$$u_{ijk}^{n+1} = {\color{red}F}(u_{ijk}^{n})$$

${\color{red}F}$: from numerics

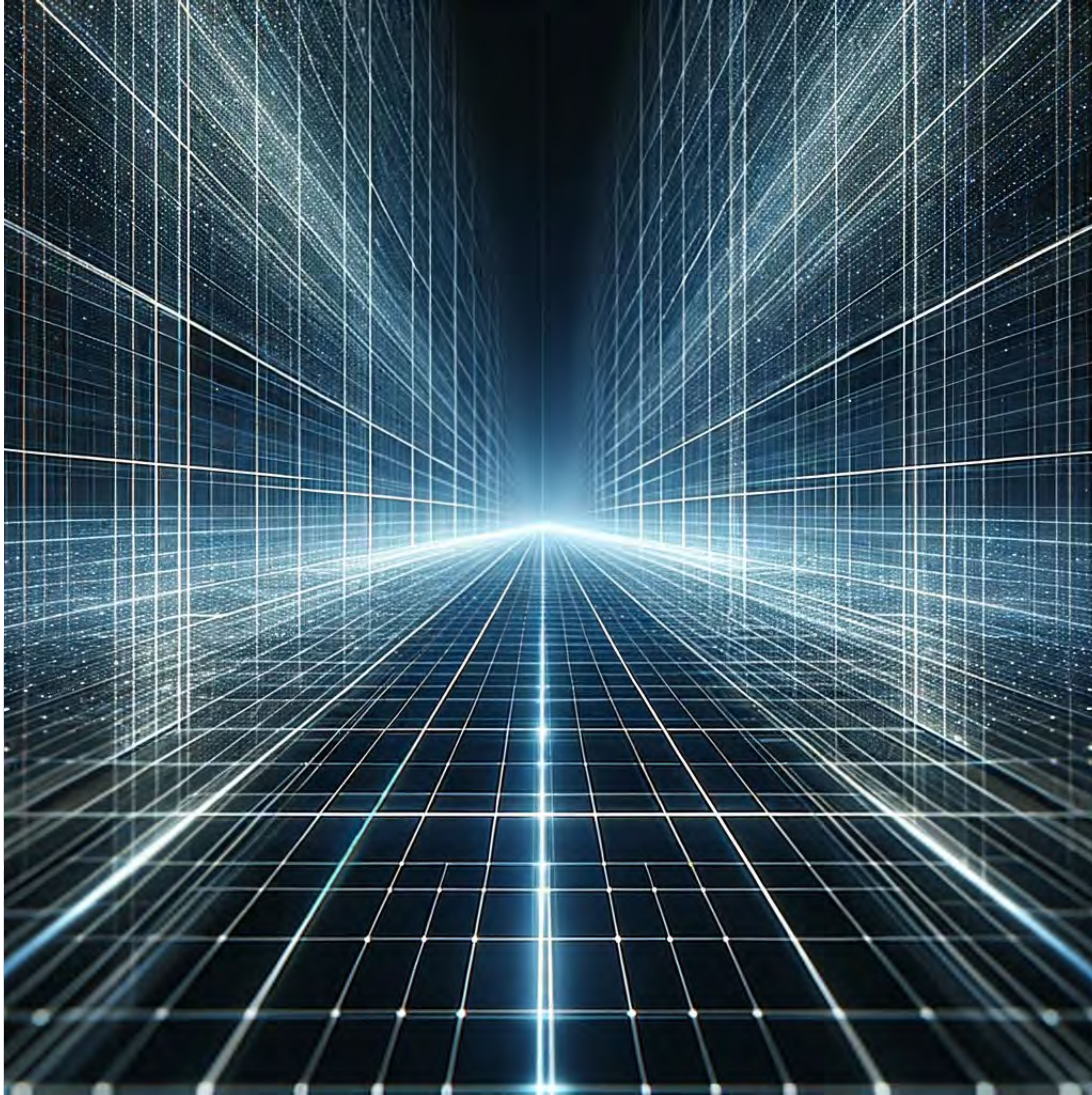
+

${\color{red}F}$: through RL


nature machine intelligence
ARTICLES
https://doi.org/10.1038/s42256-020-00272-0
Check for updates

Automating turbulence modelling by multi-agent reinforcement learning

Guido Novati[1], Hugues Lascombes de Laroussilhe[1,2] and Petros Koumoutsakos[1,3]



Flow Modeling via Reinforcement Learning

$$F(\mathbf{u}) = 0$$

FULL EQUATION - EXPENSIVE

$$\mathbf{u} \;=\; \bar{\mathbf{u}} + \mathbf{u}'$$

$\bar{\mathbf{u}}$: reduced/filtered/coarse grained $\mathbf{u}' = \mathbf{u} - \bar{\mathbf{u}}$

$$F(\bar{\mathbf{u}}) \;=\; R[\bar{\mathbf{u}}, \mathbf{u}']$$

$$F(\bar{\mathbf{u}}) \;=\; M[\bar{\mathbf{u}}]$$

LES/RANS/CGMD/...

$$F(\bar{\mathbf{u}}(\mathbf{x}, t)) \;=\; \pi[\mathbf{s}(\bar{\mathbf{u}}), \mathbf{a}(\mathbf{s})]$$

RL for LES

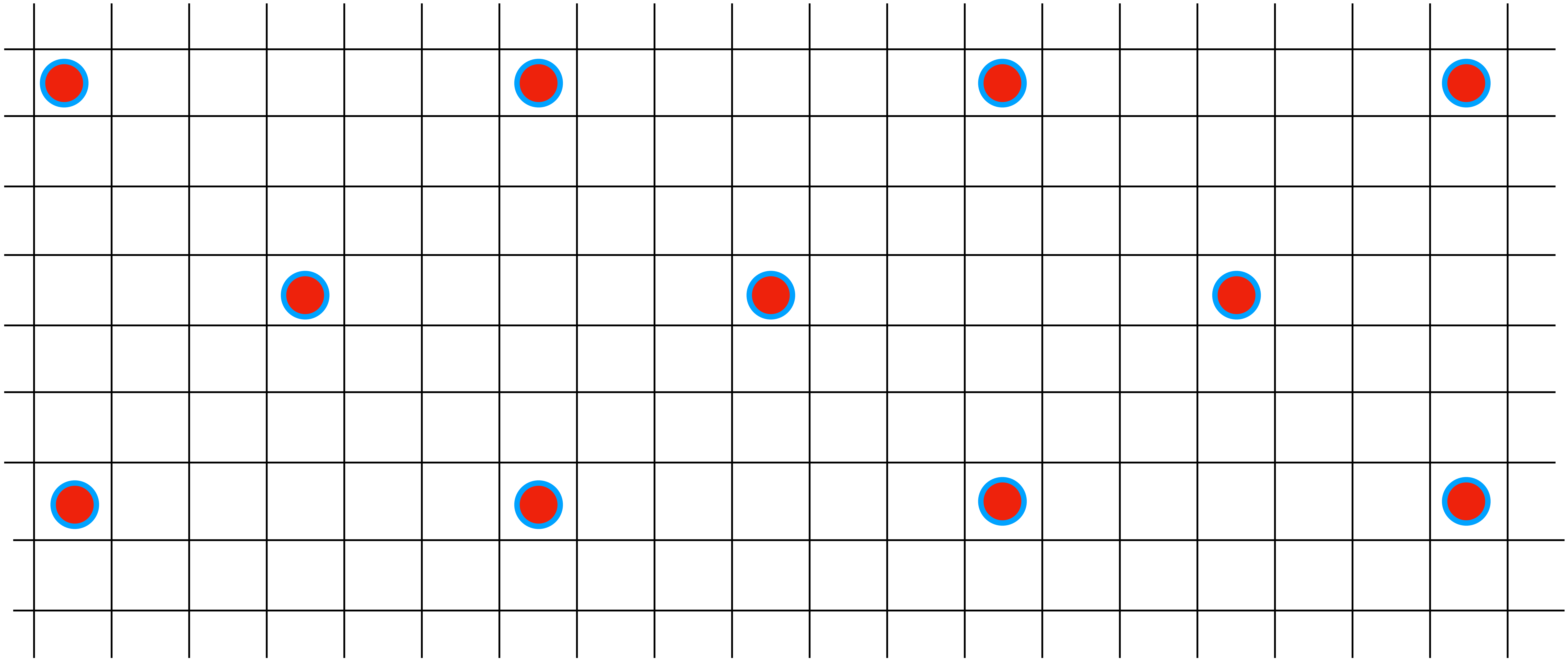
RL: agent learns to optimize its long-term consequences on the environment.

$$u_{ij}^{n+1} = F(u_{ij}^{n})$$

F : from numerics + F : through RL

RL Agents

LES Grid

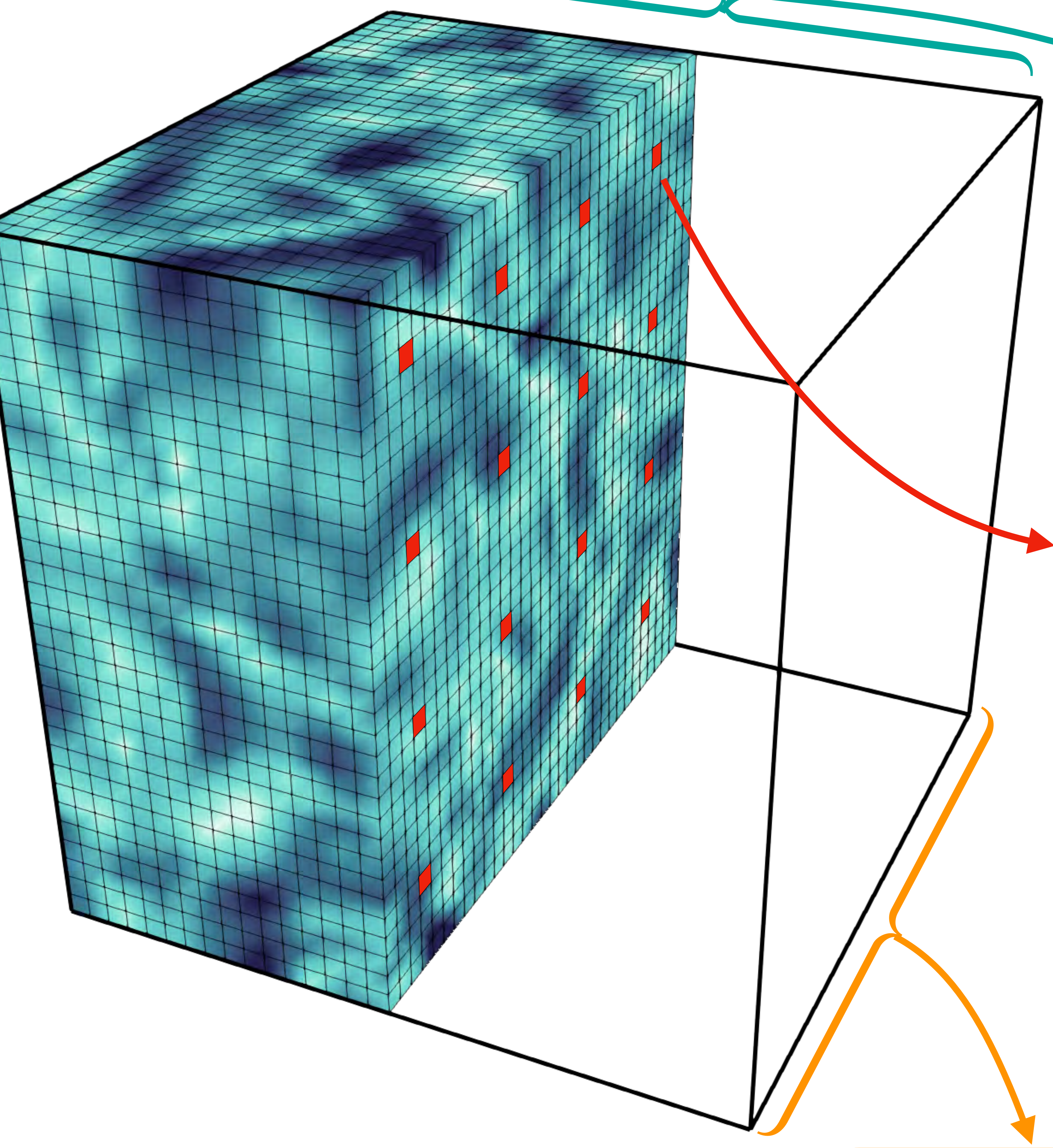


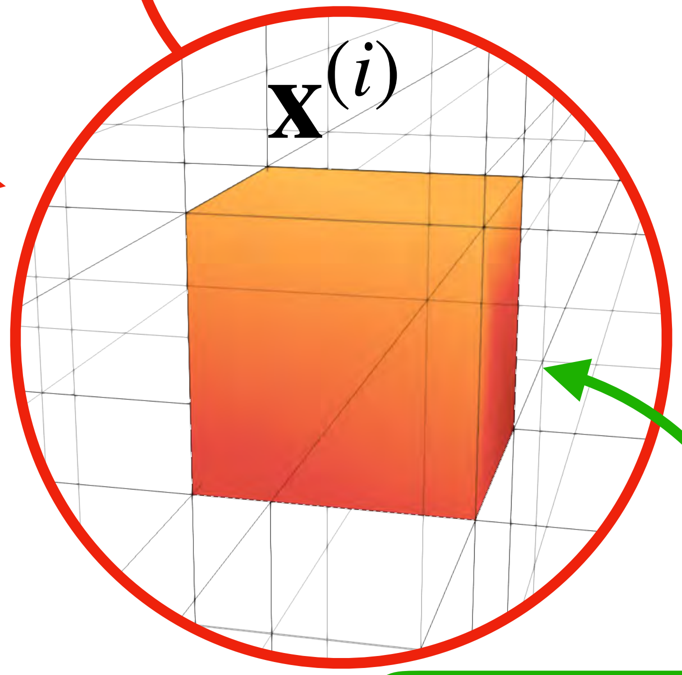
Multi-Agent Deep Reinforcement Learning

local & global state information

$$s^{grid}(t) = \left\{ \epsilon_{visc}(t),\ \epsilon_{tot}(t),\ E(k,t)\Big|_{k=1:15} \right\}$$

$$s\left(\mathbf{x}^{(i)}, t\right) = \left\{ \lambda_k^{\nabla u}\left(\mathbf{x}^{(i)}, t\right)\Big|_{k=1:5},\ \lambda_k^{\Delta u}\left(\mathbf{x}^{(i)}, t\right)\Big|_{k=1:6} \right\}$$





$$\nu_t = (C_s\Delta)^2 ||\mathbf{S}||$$

agent i

$$C_s\left(\mathbf{x}^{(i)}, t\right) = a\left(\mathbf{x}^{(i)}, t\right) \sim \pi_w\left(\cdot \mid s\left(\mathbf{x}^{(i)}, t\right)\right)$$

$$r(t) = -\mathscr{D}(E, E_{DNS})$$

- Common policy
- Agents act locally on (C_s)
- Training on multiple Re_λ

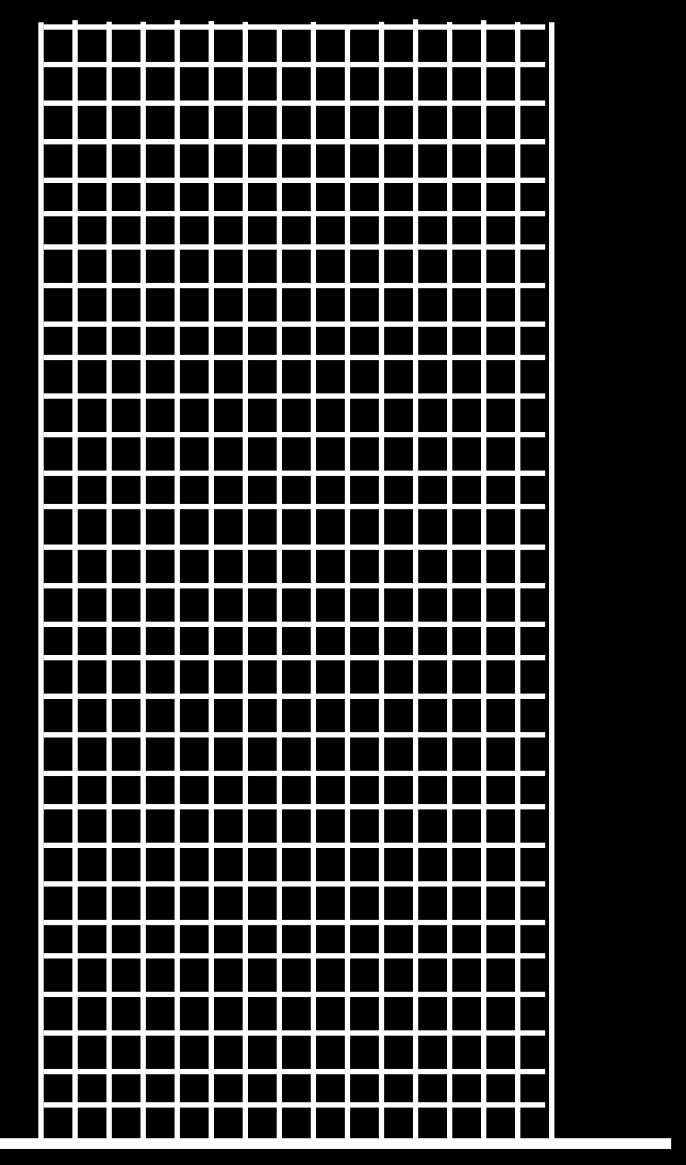
Energy spectra for DNS (solid black line)

Standard Smagorinsky Model (purple), Dynamic Smagorinsky Model (green),

MARL policy π^{LL}, MARL policy π^{G} and MARL policy π^{LL} trained exclusively from data for Re = 111

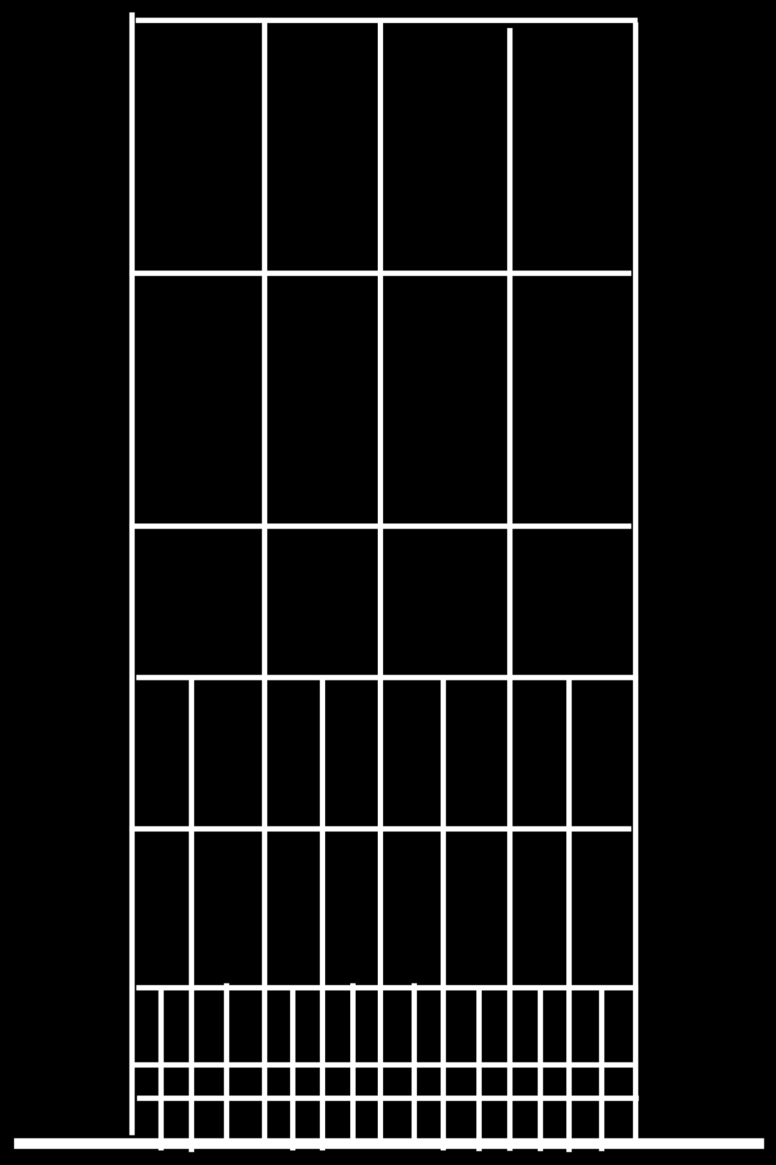
Training set: $Re_\lambda \in \{65, 76, 88, 103, 120, 140, 163\}$

DNS



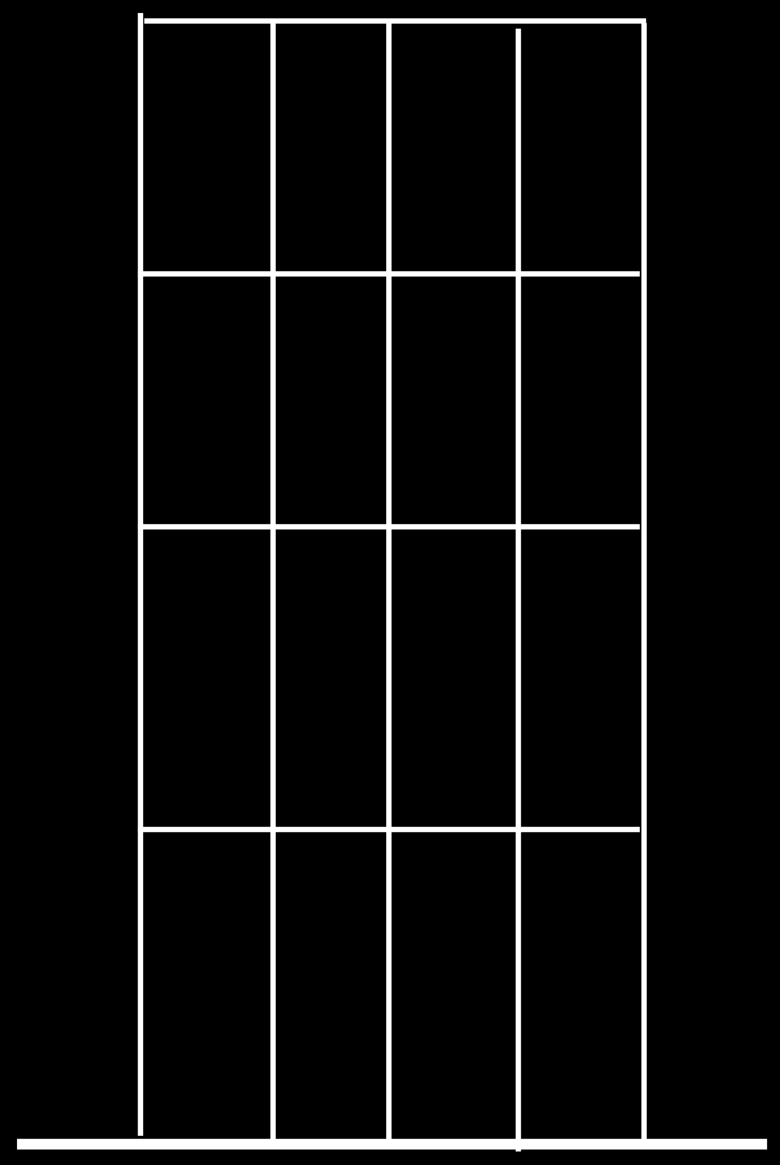
$O(Re^{2.6})$

WRLES



$O(Re^{1.9})$

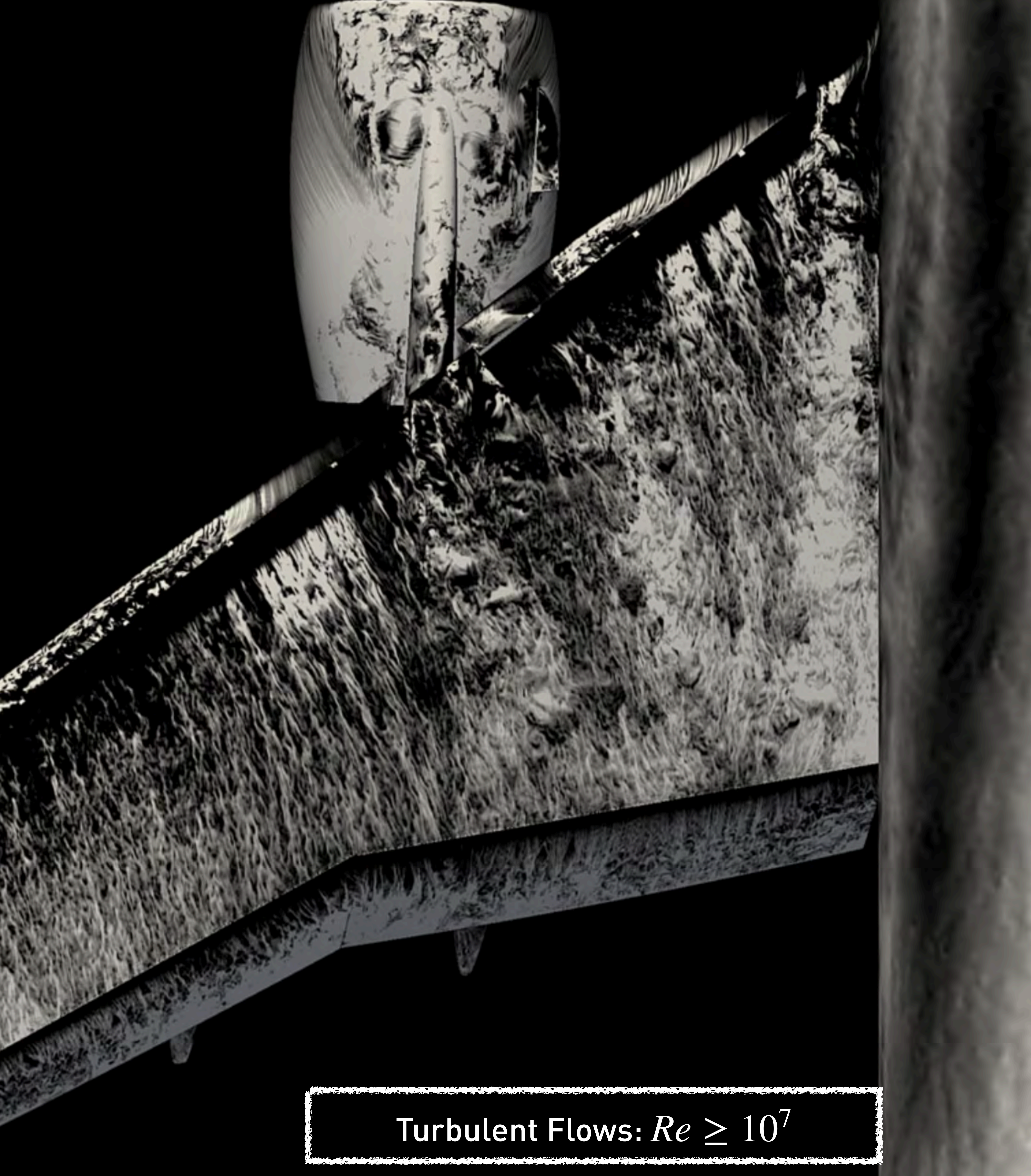
WMLES



$O(Re^{0-1})$

Chapman (1979), Choi & Moin (2011)

How many grid points ?



- LES works well for unbounded domains

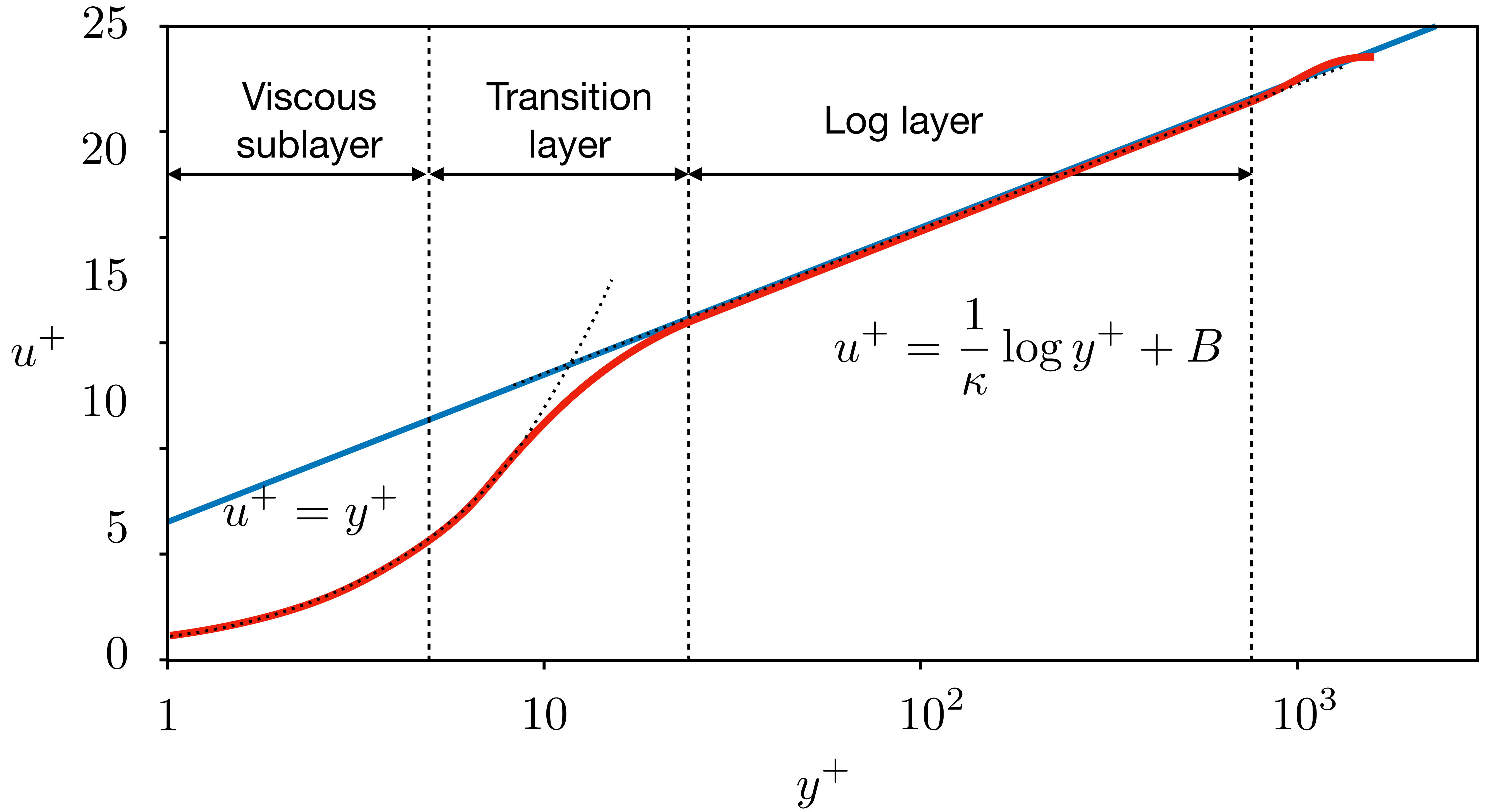
- Challenges when simulating wall bounded flows


$1mm^3$ on
an airplane wing
WALL
TURBULENCE

State-of-the-art wall models

- Algebraic wall models

$$u^+ = \frac{1}{\kappa} \log y^+ + B$$



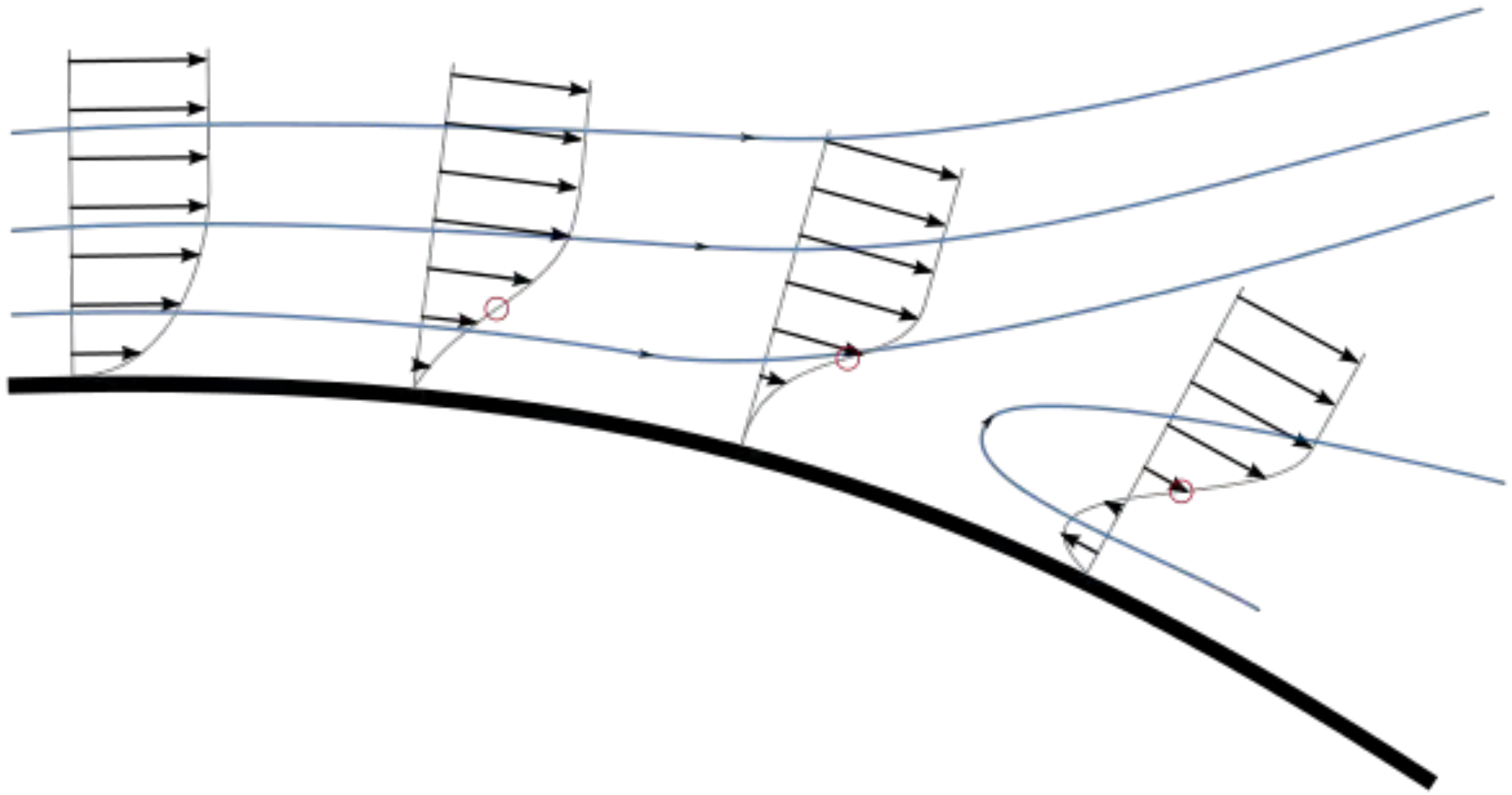
- RANS-based wall model

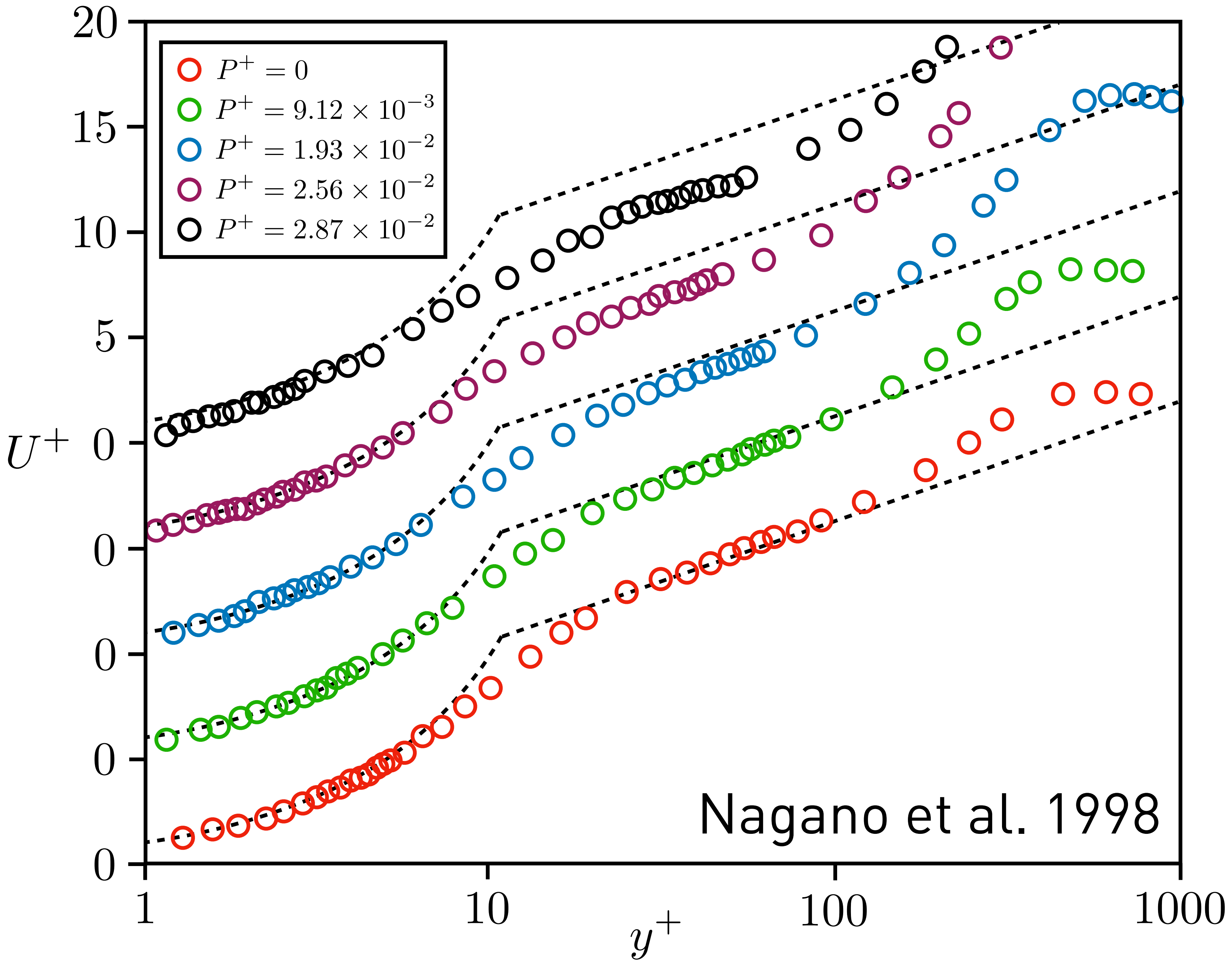
$$\frac{\mathrm{d}}{\mathrm{d}y}\left[(\nu + \nu_{wm})\frac{\mathrm{d}U_\parallel}{\mathrm{d}y}\right] = 0 \qquad \nu_{wm} = \kappa y \sqrt{\frac{\tau_w}{\rho}}[1 - \exp(-y^+/A^+)]^2$$

Deardorff (1970), Schumann (1975), Kawai & Larsson (2012) ...

State-of-the-art wall models: shortcoming

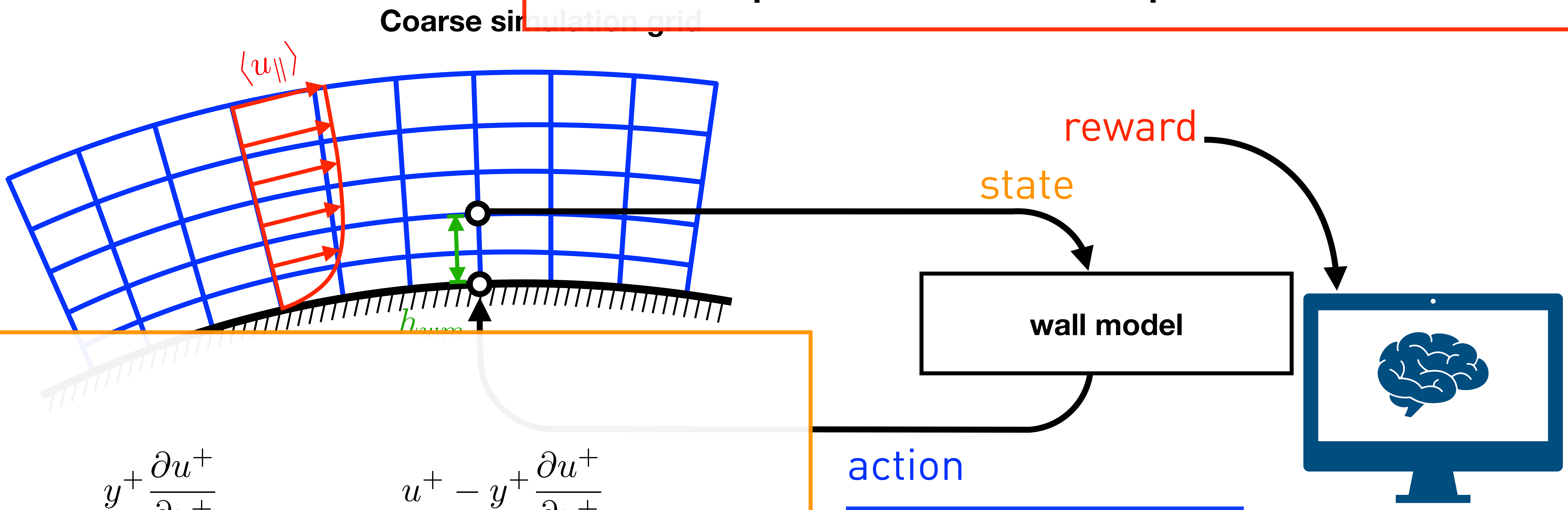
Example: adverse pressure gradient flow





Wall model needs to be recalibrated for all values of pressure-gradient

$$r(t_n) = \underbrace{\left(\frac{|\tau_w - \tau_w^m(t_n)| - |\tau_w - \tau_w^m(t_{n-1})|}{\tau_w} \right)}_{\textbf{improvement from last step}} + \underbrace{\mathbf{1}\left(\frac{|\tau_w - \tau_w^m(t_n)|}{\tau_w} < 0.01 \right)}_{\textbf{bonus if error < 1\%}}$$



$$y^+ \frac{\partial u^+}{\partial y^+}$$

Log-law slope

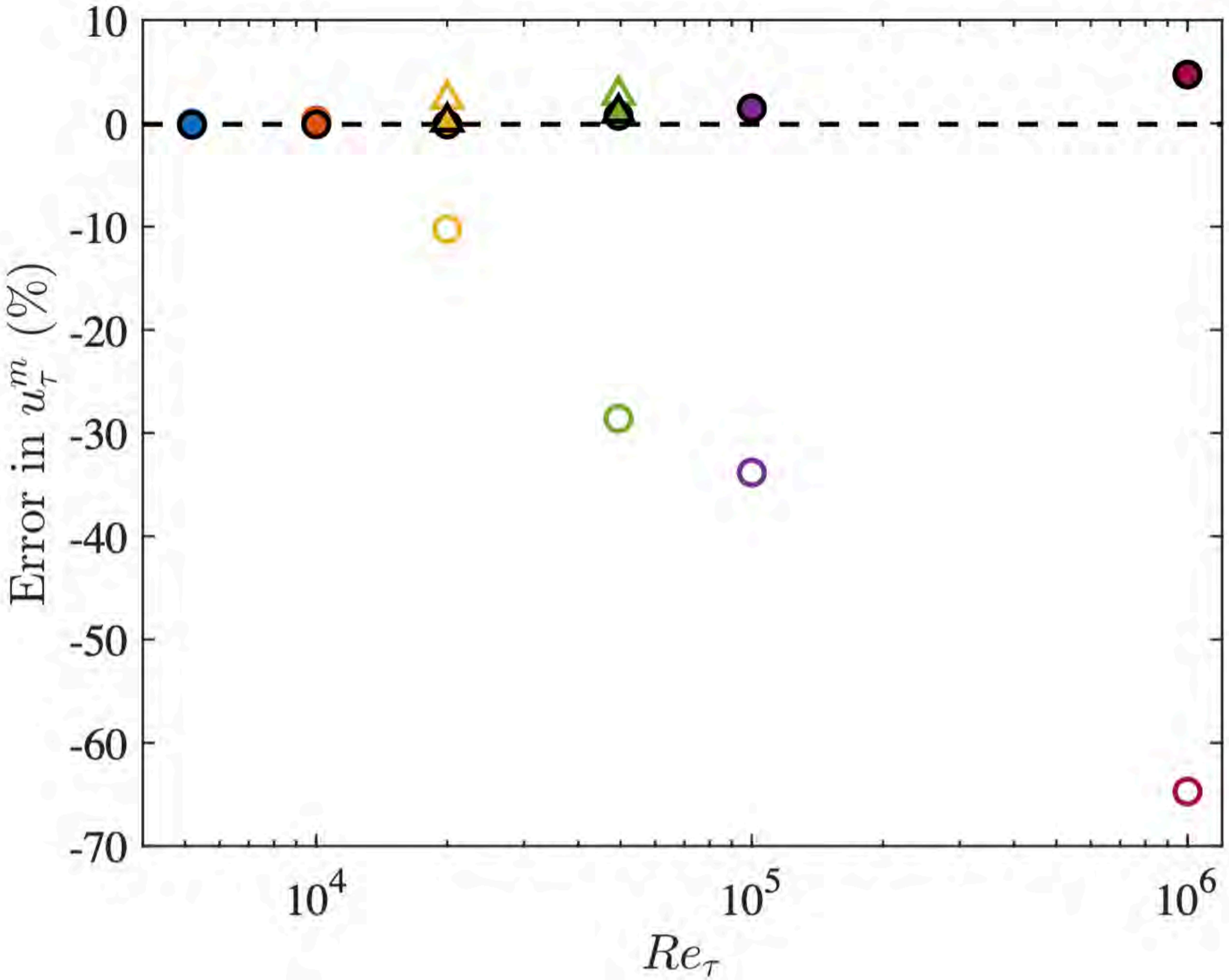
$$u^+ - y^+ \frac{\partial u^+}{\partial y^+}$$

Log-law intercept

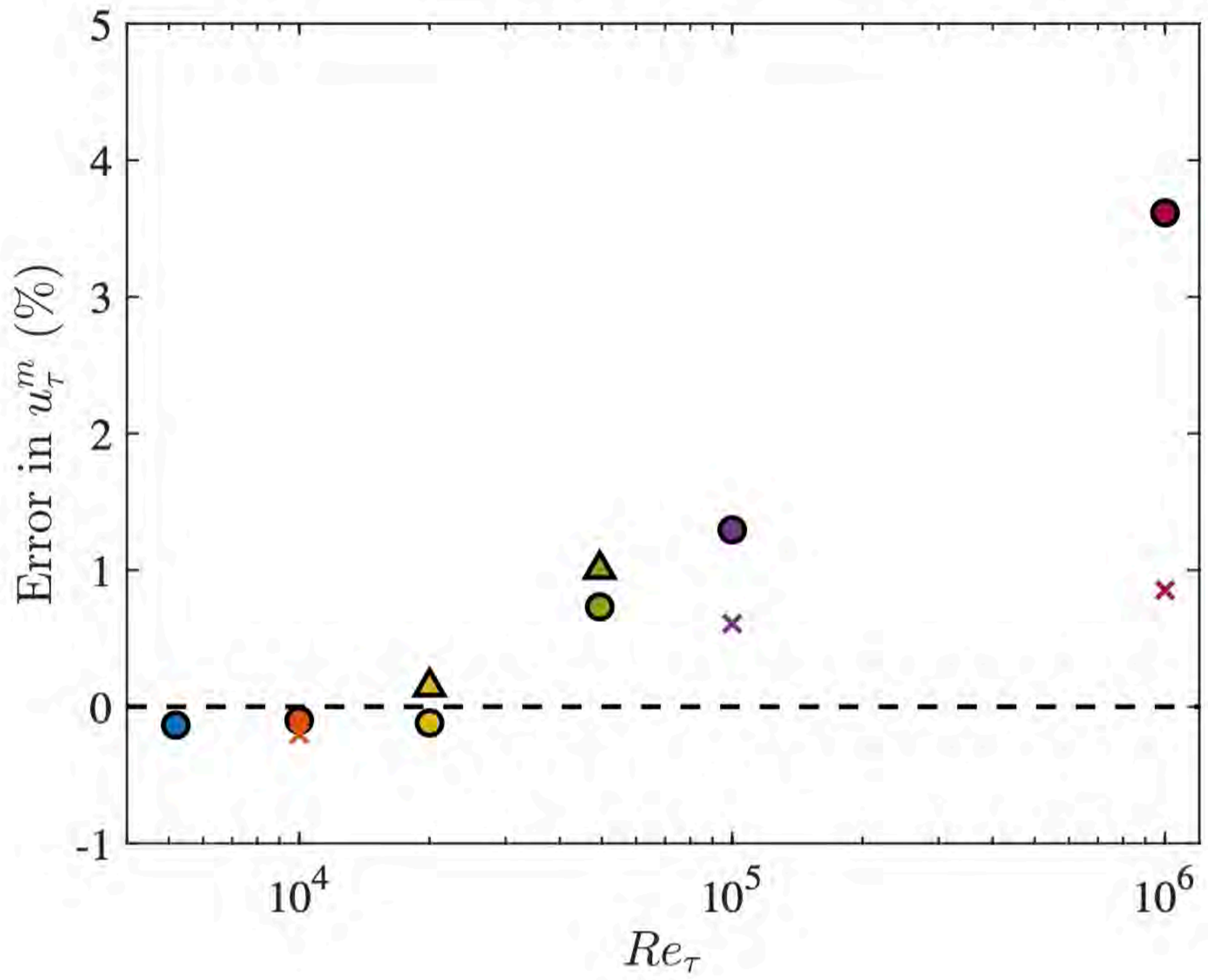
at a off-wall location h_{wm}

$$\tau_w^m(t_{n+1}) = a\tau_w^m(t_n)$$

$$a \in [0.9, 1.1]$$

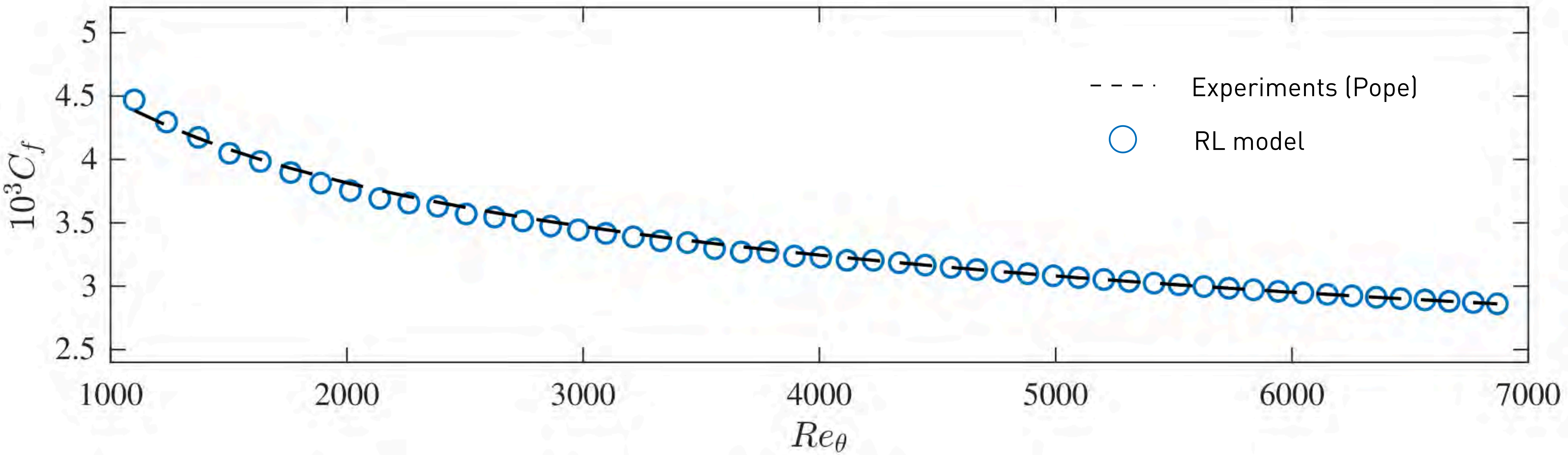


Error in time-averaged wall-shear stress obtained from the VWM (empty) and LLWM (filled) for various Reynolds numbers. Circles indicate the standard grid with Δy = 0.05 and triangles indicate refined cases.



(b) Zoomed in version of (a) for LLWM with error in EQMW (crosses) for three Reynolds numbers.

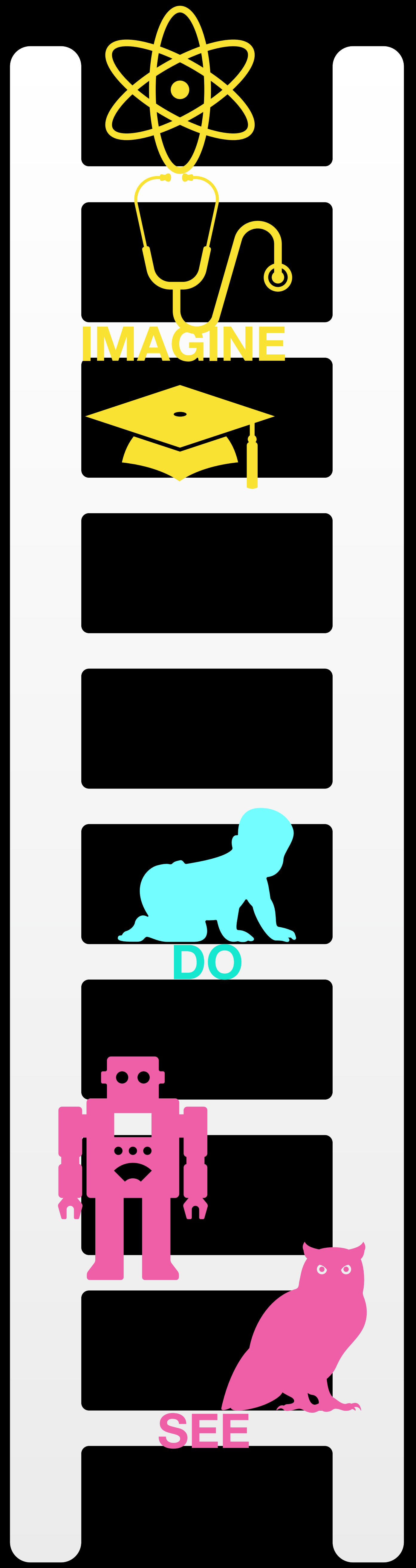
TESTING II: Evolving turbulent boundary layer



nature COMMUNICATIONS
ARTICLE
https://doi.org/10.1038/s41467-022-28957-7 OPEN

Scientific multi-agent reinforcement learning for wall-models of turbulent flows

H. Jane Bae [1,2] & Petros Koumoutsakos [1,3]



THE LADDER OF CAUSATION

(adapted from the “Book of Why” by J. Pearl)

III. COUNTER FACTUALS

ACT: Imagining, Retrospection, Understanding
ASK: *What if I had done … ? Why ?*
Did X cause Y ? What if I acted differently ?
What if X had not occurred ?

EXAMPLE:
Was it the aspirin that stopped my headache ?

II. INTERVENTION

ACT: Doing, Intervening
ASK: *What if I do … ? How ?*
How can I make D ?
How would D be if I do H ?

EXAMPLE:
What if I take aspirin ?

I. ASSOCIATION

ACT: Seeing, Observing
ASK: *What if I see … ?*
How are variables related ?
How would seeing D change my belief in H ?

EXAMPLE:
I sneeze -did I catch a cold ?

Interpret or Explain?

https://docs.aws.amazon.com/whitepapers/latest/model-explainability-aws-ai-ml/interpretability-versus-explainability.html

- Interpretability — high model transparency and understanding
- **exactly why and how** the model is generating predictions, as we can observe the inner mechanics of the AI/ML method.
- This leads to **interpreting the model's weights and features to determine the given output.**

- **Explainability** — how to take an ML model and explain the behavior in human terms.
- For complex models one cannot understand how and why the inner mechanics impact the prediction.
- model agnostic methods may discover **meaning between input data attributions and model outputs,** which enables explaining the nature and behavior of the AI/ML model.

Machine Learning

Should AI Models Be Explainable? That depends.

A Stanford researcher advocates for clarity about the different types of interpretability and the contexts in which it is useful.

But AI models do not need to be interpretable to be useful, says **Nigam Shah**, professor of medicine (biomedical informatics) and of biomedical data science at Stanford University and an affiliated faculty member of the **Stanford Institute for Human-Centered Artificial Intelligence**. That's especially true in medicine, he says, where doctors routinely offer treatments without knowing how or why they work.

"Of the 4,900 drugs prescribed on a routine basis, we don't fully know how most of them really work," says Shah. "But we still use them because we have convinced ourselves via randomized control trials that they are beneficial."

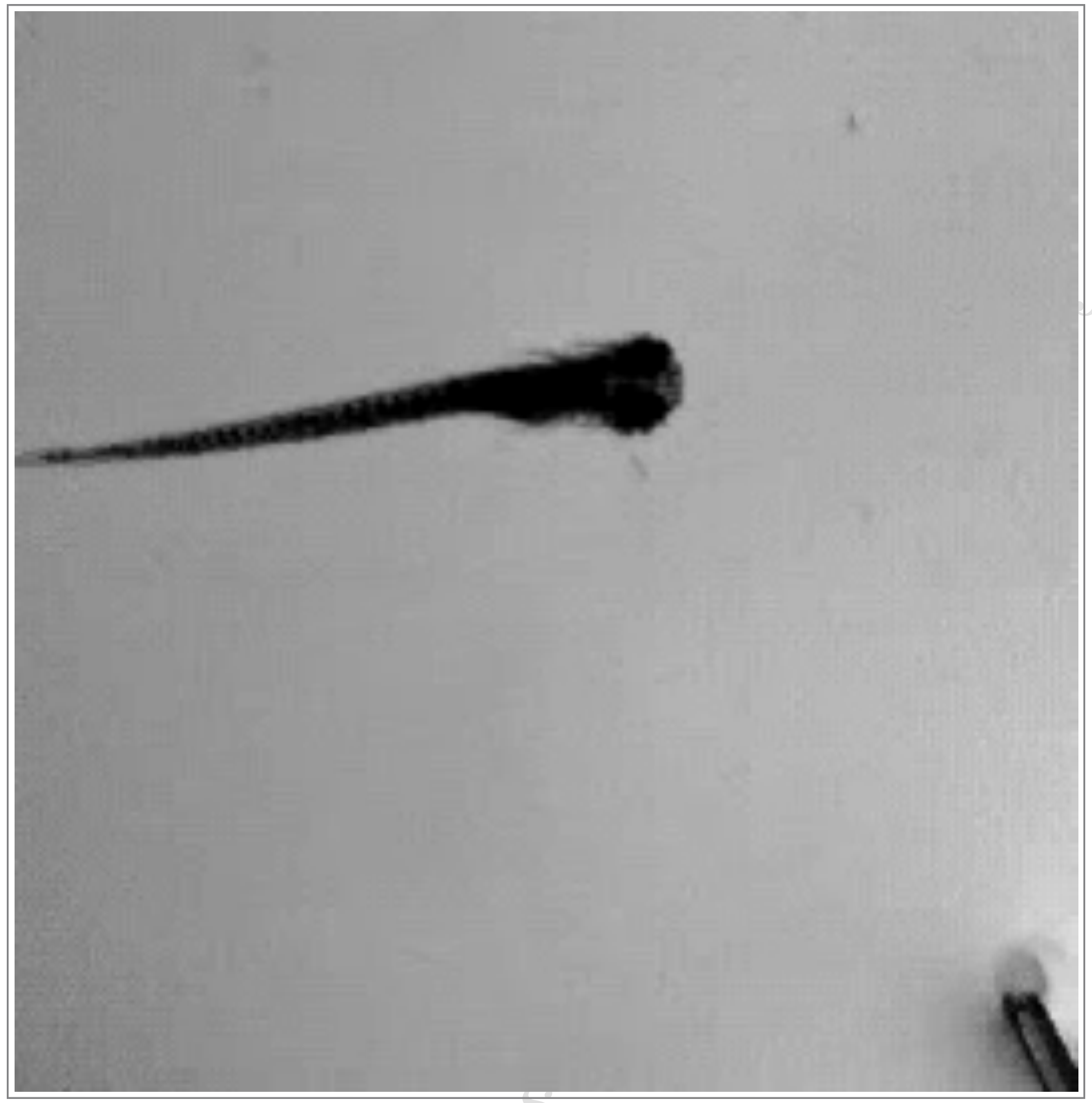
TECHNOLOGY FEATURE | 09 January 2023

The reproducibility issues that haunt health-care AI

Health-care systems are rolling out artificial-intelligence tools for diagnosis and monitoring. But how reliable are the models?

Emily Sohn

Explaining through Uncertainty Quantification



$$\int_{\mathcal{Y}}\int_{\mathcal{R}} \log \frac{p(\boldsymbol{y}|\boldsymbol{r},\boldsymbol{s})}{p(\boldsymbol{y}|\boldsymbol{s})}\, p(\boldsymbol{r})\, p(\boldsymbol{y}|\boldsymbol{r},\boldsymbol{s})\, \mathrm{d}\boldsymbol{r}\, \mathrm{d}\boldsymbol{y}$$

WHERE TO MEASURE ?

Bayesian Optimal Experimental Design

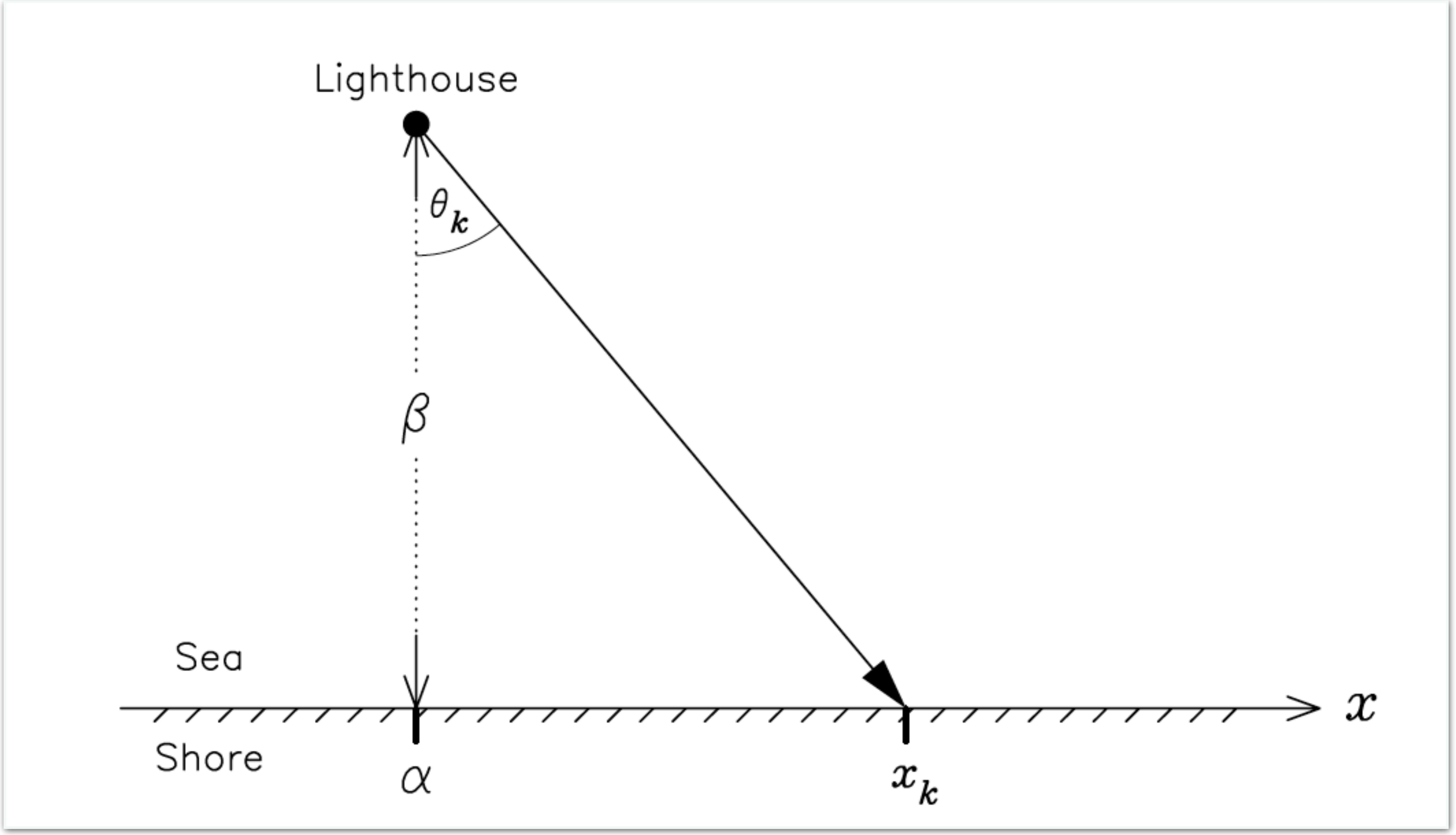
$$\hat{U}(\boldsymbol{s}) = \frac{1}{N_{\mathcal{D}}}\sum_{i=1}^{N_r}\sum_{j=1}^{N_{\mathcal{D}}} w_i p(\boldsymbol{r}^{(i)}) \left[\log p(\boldsymbol{y}^{(i,j)}|\boldsymbol{r}^{(i)},\boldsymbol{s}) - \log\left(\sum_{k=1}^{N_r} w_k p(\boldsymbol{r}^{(k)}) p(\boldsymbol{y}^{(i,j)}|\boldsymbol{r}^{(k)},\boldsymbol{s})\right)\right]$$

THE LIGHTHOUSE PROBLEM [Gull1988]

Lighhouse signals, emitted at random intervals, are recorded on a straight coastline.

Where is the Lighthouse ?

THE LIGHTHOUSE PROBLEM [Gull1988]



"A lighthouse is somewhere off a piece of **straight coastline** at a **position α along the shore** and a **distance β out at sea.** It emits a series of short highly collimated flashes at **random intervals and hence at random azimuths**.

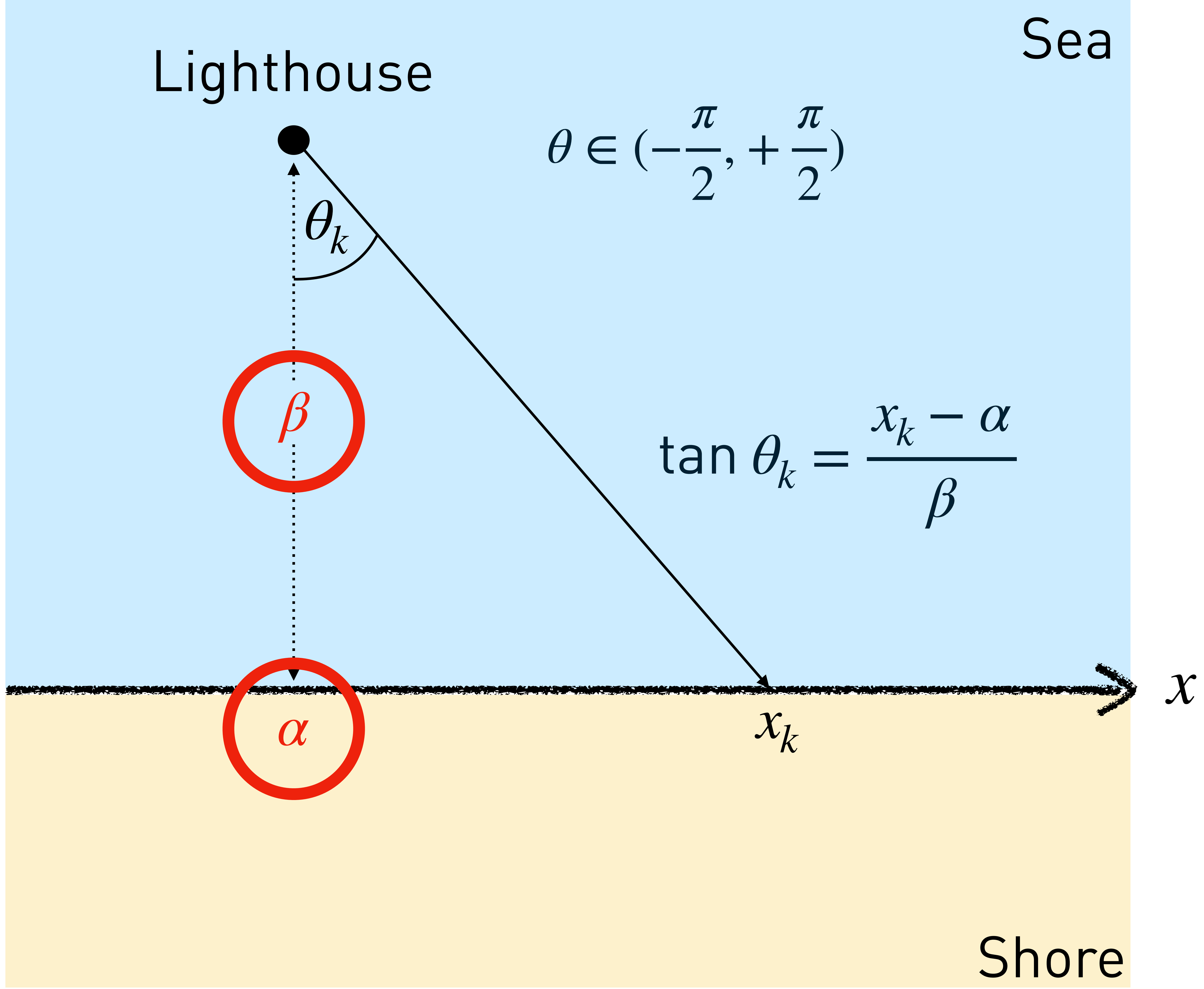
These pulses are intercepted on the coast by photo-detectors that record only the fact that a flash has occurred, but not the angle from which it came. N flashes have so far been **recorded at positions $\{x_k\}$**.

Where is the lighthouse?" [Sivia2006]

A BAYES APPROACH

$$p(\alpha,\beta\,|\,x_k,\mathscr{M}) = p(x_k\,|\,\alpha,\beta,\mathscr{M}) \times p(\alpha,\beta,\mathscr{M})$$

Likelihood Prior



Assumptions

$$\theta \sim \mathscr{U}(-\frac{\pi}{2},\frac{\pi}{2})$$

$$\beta \in (0, \beta_{max})$$

$$\alpha \in (-\alpha_{min}, \alpha_{max})$$

Likelihood $p(\theta_k\,|\,\alpha,\beta) = \frac{1}{\pi}$

change of variables

$$p(x_k\,|\,\alpha,\beta,\mathscr{M}) = p(\theta_k\,|\,\alpha,\beta)\left|\frac{d\theta}{dx}\right| = \frac{\beta}{\pi[\beta^2 + (x_k - \alpha^2)]}$$

I - Assume known distance from shore (β_0)

Unknown to estimate: position along shore-> α

FIND: Posterior pdf of α using Bayes' theorem: Posterior ~ Likelihood x Prior

Likelihood of observed data (assume i.i.d) $p(\{x_k\}\,|\,\alpha, \beta_0, \mathcal{M}) = \prod_{k=1}^{N} p(x_k\,|\,\alpha, \beta_0, \mathcal{M})$

Prior

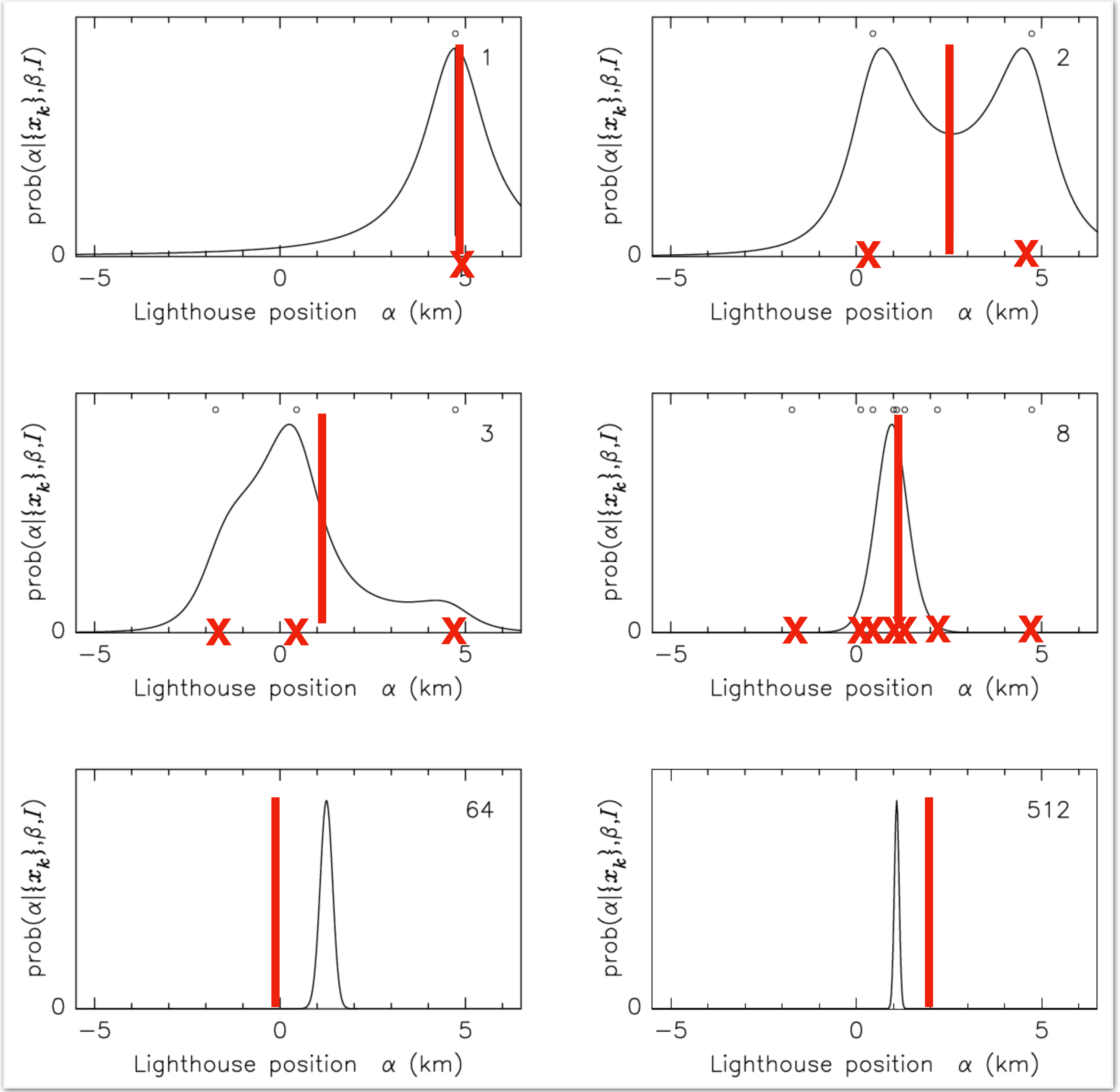
$$p(\alpha\,|\,\beta_0, \mathcal{M}) = \begin{cases} A, & \text{if } \alpha_{min} \leq \alpha \leq \alpha_{max} \\ 0, & \text{otherwise} \end{cases}$$

Posterior ~ Likelihood x Prior

$$p(\alpha\,|\,\{x_k\}, \beta_0, \mathcal{M}) \propto p(\{x_k\}\,|\,\alpha, \beta_0, \mathcal{M}) \times p(\alpha\,|\,\beta_0, \mathcal{M})$$

$\rightarrow$ find $\hat{\alpha}_{MAP} = \text{argmax}\, p(\alpha\,|\,\{x_k\}, \beta_0, \mathcal{M})$

I - known distance from shore (β_0)



Sample mean is not always a useful number

—

let the posterior pdf decide what is best

data generated with $\alpha^* = 1.0$ and $\beta^* = 1.0$

Figure from [Sivia2006]

II - position & distance unknown

Posterior pdf of **α** & **β** using Bayes' theorem:

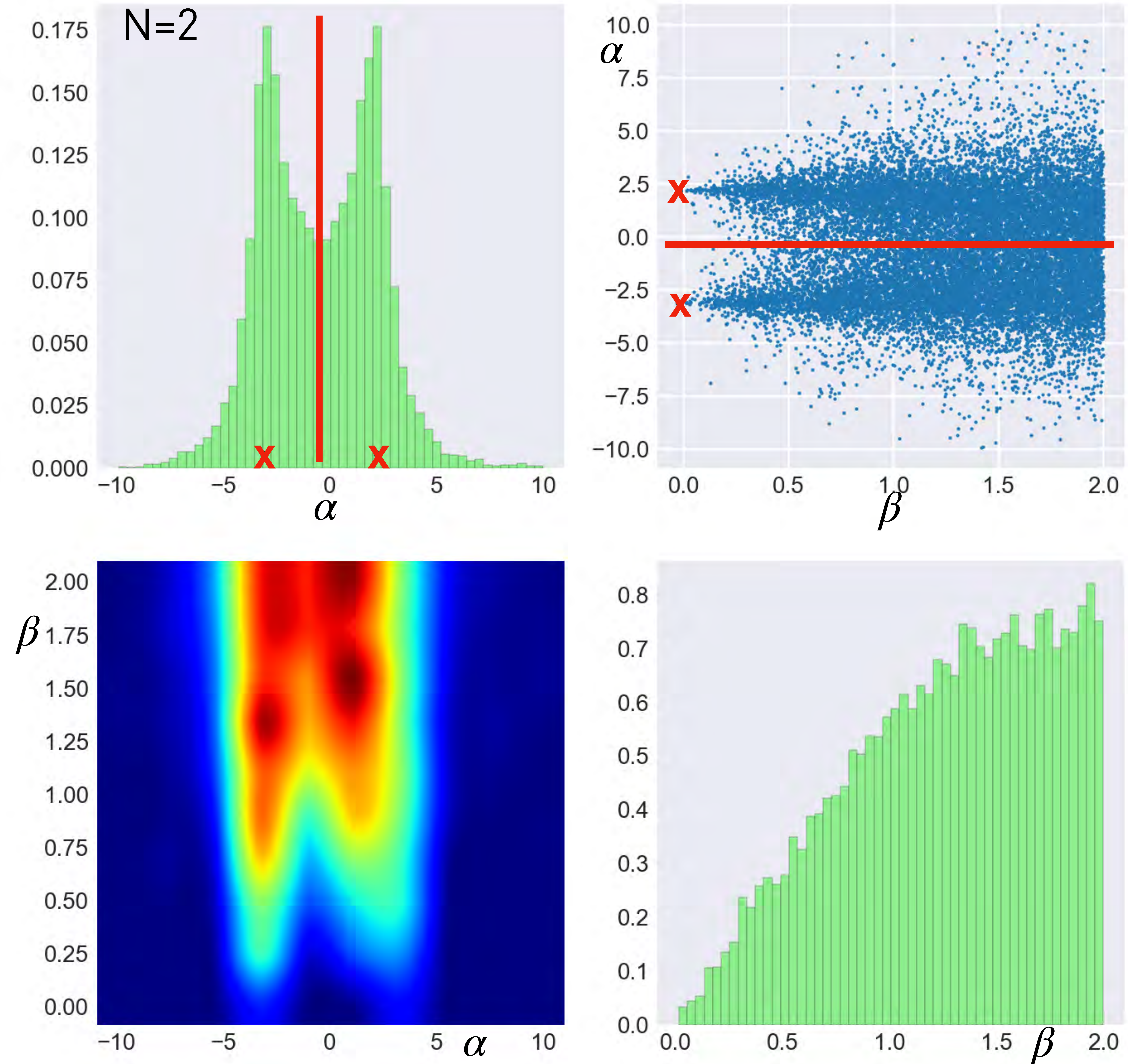
$$p(\alpha, \beta \,|\, \{x_k\}, \mathscr{M}) \propto p(\{x_k\} \,|\, \alpha, \beta_0, \mathscr{M}) \times p(\alpha \,|\, \mathscr{M}) \times p(\beta \,|\, \mathscr{M})$$

$$p(\alpha \,|\, \mathscr{M}) = \begin{cases} A, & \text{if } \alpha_{min} \leq \alpha \leq \alpha_{max} \\ 0, & \text{otherwise} \end{cases}$$

$$p(\beta \,|\, \mathscr{M}) = \begin{cases} B & \text{if } 0 \leq \beta \leq \beta_{max} \\ 0, & \text{otherwise} \end{cases}$$

$$p(\alpha \,|\, \{x_k\}, \mathscr{M}) = \int p(\alpha, \beta \,|\, \{x_k\}, \mathscr{M}) \mathrm{d}\beta$$

→ find posterior & marginals of α and β through sampling

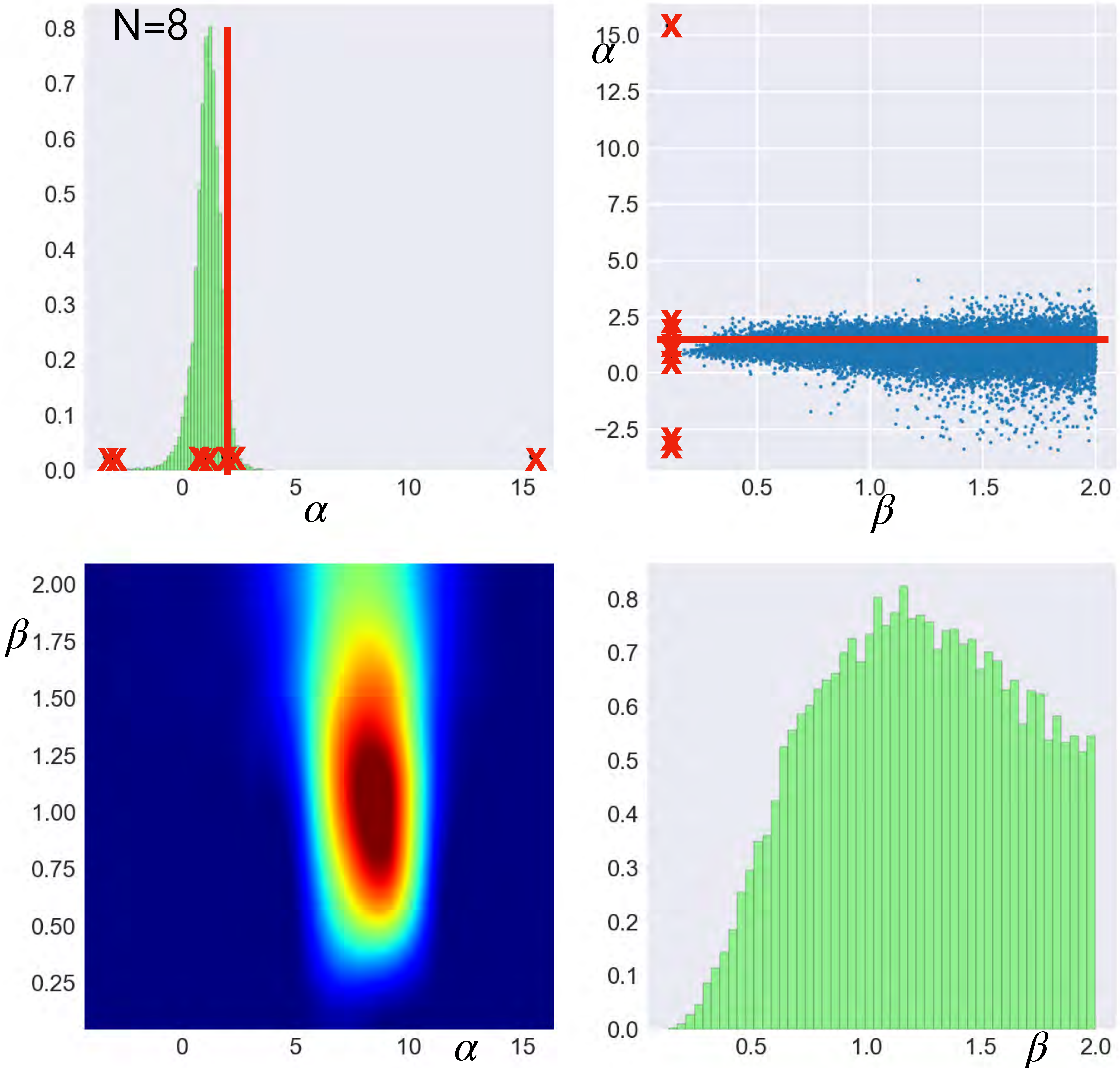


II - position & distance unknown

Sampling Posterior
and
Marginal Distribution

data generated with $\alpha^* = 1.0$ and $\beta^* = 1.0$

$\alpha_{min} = -10.0, \alpha_{max} = +10.0$ and $\beta_{max} = 2.0$



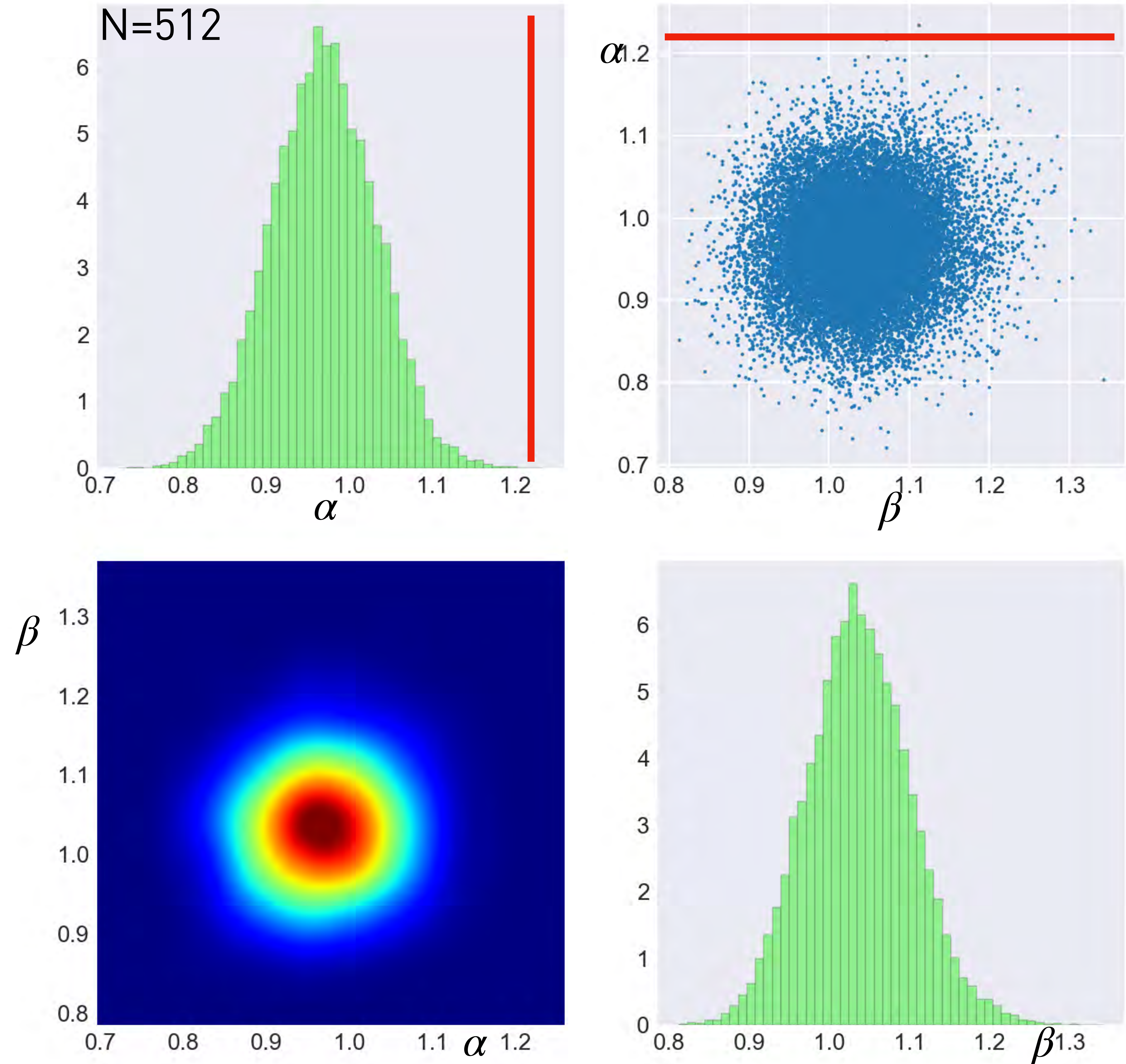
II - position & distance unknown

Sampling Posterior
and
Marginal Distribution

data generated with $\alpha^* = 1.0$ and $\beta^* = 1.0$

$\alpha_{min} = -10.0, \alpha_{max} = +10.0$ and $\beta_{max} = 2.0$

II - position & distance unknown



Sampling Posterior
and
Marginal Distribution

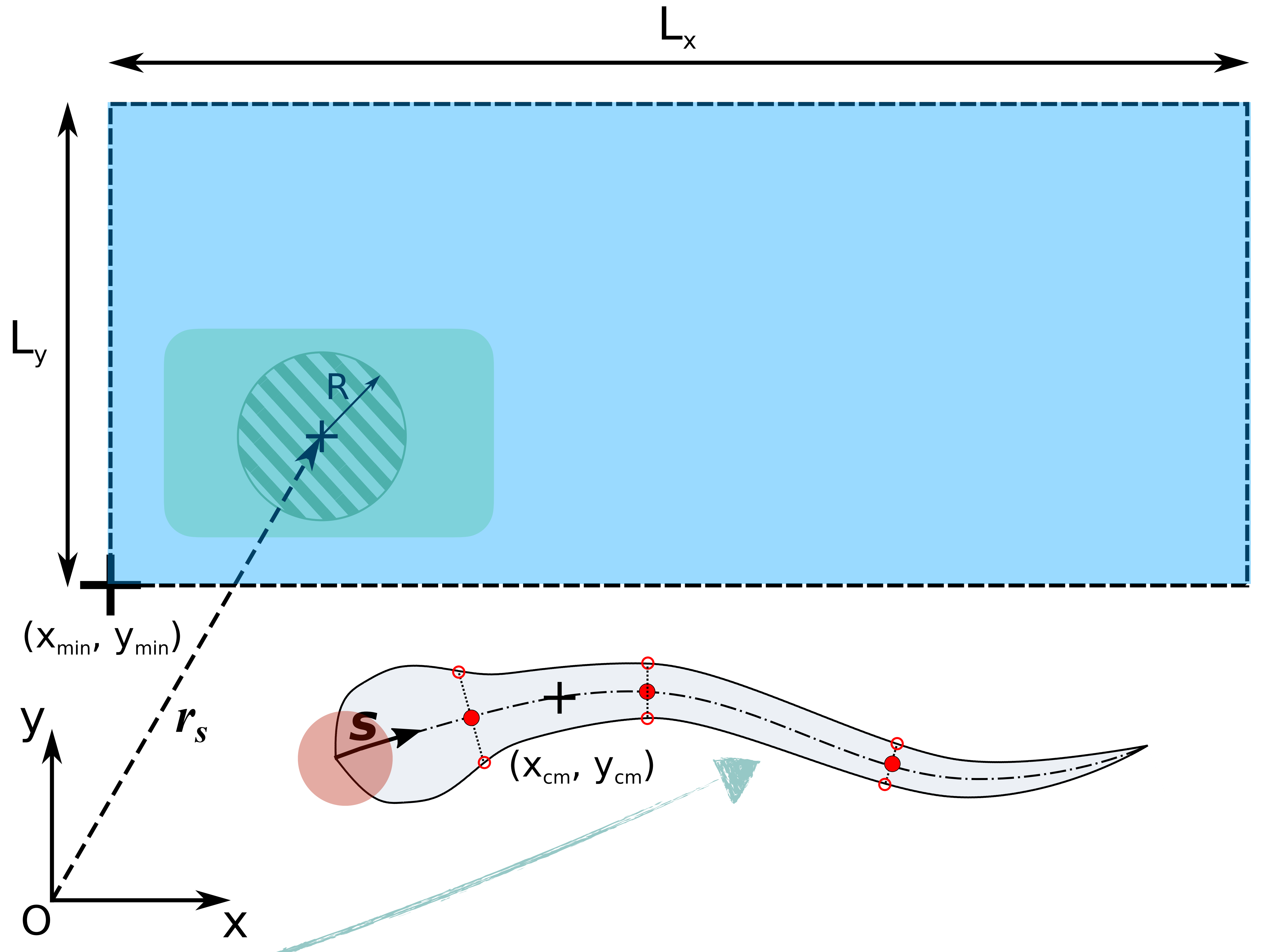
data generated with $\alpha^* = 1.0$ and $\beta^* = 1.0$

$\alpha_{min} = -10.0, \alpha_{max} = +10.0$ and $\beta_{max} = 2.0$

Optimal Sensors for Fish

Location of the disturbance $\boldsymbol{r}$

Prior belief of the disturbance $p(\boldsymbol{r})$



Measurements on the skin $\boldsymbol{y}$

Sensor Location $\boldsymbol{s}$

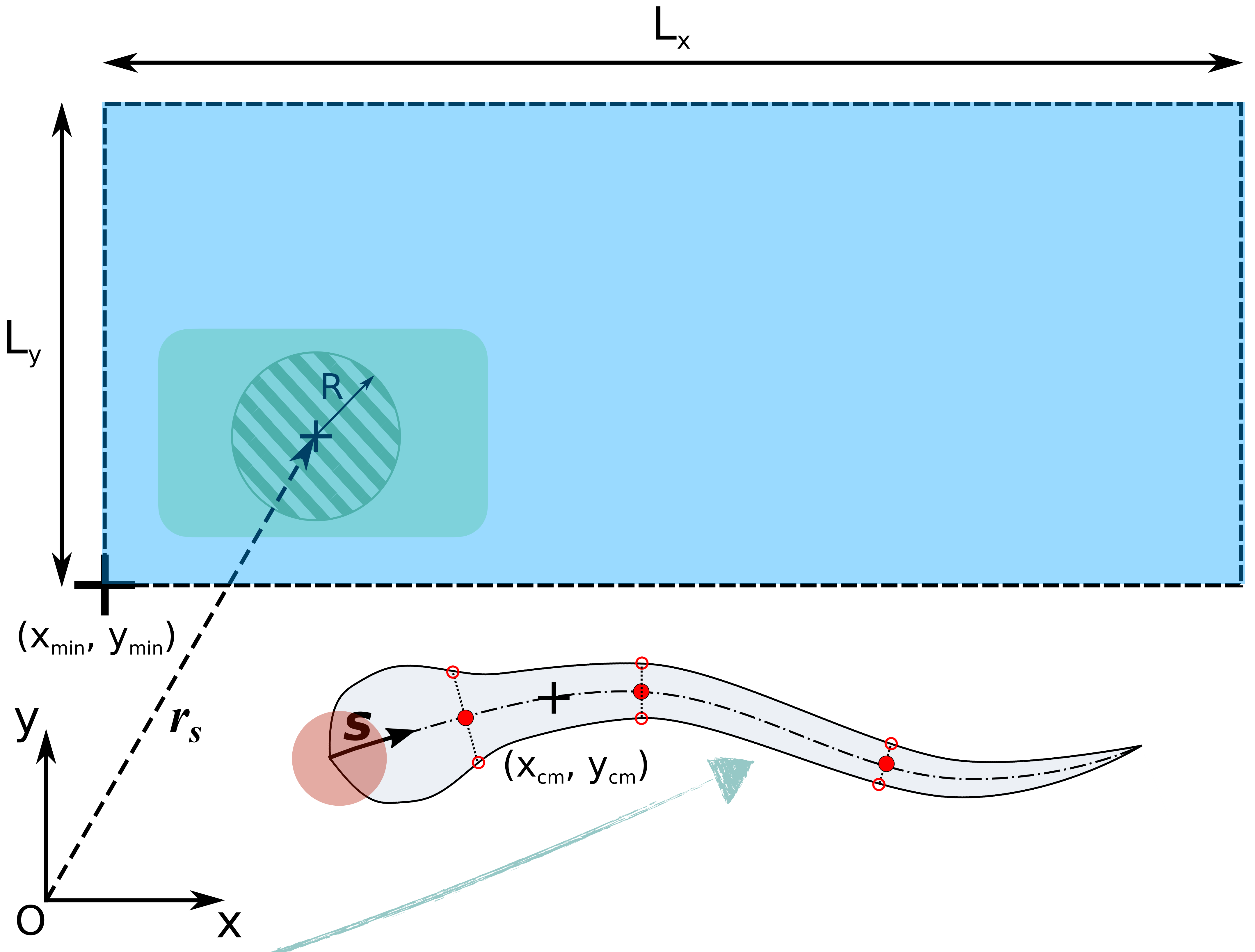
- **Assumption**: measurements come from a computational model + noise

$$\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{r}; \boldsymbol{s}) + \varepsilon$$

Optimal Sensors for Any problem

Unknown quantity

$\boldsymbol{r}$



Prior belief of for the value of r

$p(\boldsymbol{r})$

Measurements

$\boldsymbol{y}$

Sensor Location

$\boldsymbol{s}$

- **Assumption**: measurements come from a computational model + noise

$$\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{r};\boldsymbol{s}) + \varepsilon$$

Experimental Design

- Perform an experiment in order to estimate a set of parameters r
- The experiment depends on a set of design variables s
- Performing the experiment we observe a quantity y
- **How to chose the experimental design variables (here the sensors) ?**

➡ Maximize the information gained by the experiment over any prior information

Expected Utility

- **"Distance" of posterior from prior:**

$$D_{KL}(p(\boldsymbol{r} \,|\, \boldsymbol{y}, \boldsymbol{s}) \| p(\boldsymbol{r})) = \int_{\mathscr{R}} p(\boldsymbol{r} \,|\, \boldsymbol{y}, \boldsymbol{s}) \log \frac{p(\boldsymbol{r} \,|\, \boldsymbol{y}, \boldsymbol{s})}{p(\boldsymbol{r})} \mathrm{d}\boldsymbol{r}$$

- Average over all measurements:

$$U(\boldsymbol{s}) = \mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{s})} \left[D_{KL}(p(\boldsymbol{r} \,|\, \boldsymbol{y}, \boldsymbol{s}) \| p(\boldsymbol{r})) \right]$$

$$p(\boldsymbol{r}|\boldsymbol{y}, \boldsymbol{s}) = \frac{p(\boldsymbol{y}|\boldsymbol{r}, \boldsymbol{s}) p(\boldsymbol{r})}{p(\boldsymbol{y}|\boldsymbol{s})}$$

- Using Bayes' theorem:

$$U(\boldsymbol{s}) = \int_{\mathscr{Y}} \int_{\mathscr{R}} \log \frac{p(\boldsymbol{y} \,|\, \boldsymbol{r}, \boldsymbol{s})}{p(\boldsymbol{y} \,|\, \boldsymbol{s})} p(\boldsymbol{r}) p(\boldsymbol{y} \,|\, \boldsymbol{r}, \boldsymbol{s}) \mathrm{d}\boldsymbol{r} \mathrm{d}\boldsymbol{y}$$

- Optimal design:

$$\boldsymbol{s}^\star = \underset{\boldsymbol{s}}{\operatorname{argmax}}\, U(\boldsymbol{s})$$

Estimator for Expected Utility

$$U(\boldsymbol{s}) = \int_{\mathcal{Y}} \int_{\mathcal{R}} \log \frac{p(\boldsymbol{y}|\boldsymbol{r},\boldsymbol{s})}{p(\boldsymbol{y}|\boldsymbol{s})} p(\boldsymbol{r}) \, p(\boldsymbol{y}|\boldsymbol{r},\boldsymbol{s}) \, \mathrm{d}\boldsymbol{r} \, \mathrm{d}\boldsymbol{y}$$

- Approximate "distance" integral using quadrature rule (2D - few points):

$$U(s) \approx \sum_{i=1}^{N_r} w_i p\left(r^{(i)}\right) \int_{\mathcal{Y}} \log \frac{p\left(\boldsymbol{y} \,|\, \boldsymbol{r}^{(i)}, \boldsymbol{s}\right)}{p(\boldsymbol{y} \,|\, \boldsymbol{s})} p\left(\boldsymbol{y} \,|\, \boldsymbol{r}^{(i)}, \boldsymbol{s}\right) \mathrm{d}\boldsymbol{y}$$

- Estimate the "sensor" integral using Monte Carlo:

$$U(s) \approx \sum_{i=1}^{N_r} \sum_{j=1}^{N_i} \frac{w_i p\left(\boldsymbol{r}^{(i)}\right)}{N_{\mathscr{D}}} \left[\log p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{r}^{(i)}, \boldsymbol{s}\right) - \log p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{s}\right)\right]$$

- Approximate the last term with quadrature of a 2D integral:

$$p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{s}\right) = \int_{\mathscr{R}} p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{r}, \boldsymbol{s}\right) p(\boldsymbol{r}) \mathrm{d}\boldsymbol{r} \approx \sum_{k=1}^{N_r} w_k p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{r}^{(k)}, \boldsymbol{s}\right) p\left(\boldsymbol{r}^{(k)}\right)$$

- The estimator:

$$\hat{U}(\boldsymbol{s}) = \frac{1}{N_{\mathscr{D}}} \sum_{i=1}^{N_r} \sum_{j=1}^{N_{\mathscr{D}}} w_i p\left(\boldsymbol{r}^{(i)}\right) \left[\log p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{r}^{(i)}, \boldsymbol{s}\right) - \log\left(\sum_{k=1}^{N_r} w_k p\left(\boldsymbol{r}^{(k)}\right) p\left(\boldsymbol{y}^{(i,j)} \,|\, \boldsymbol{r}^{(k)}, \boldsymbol{s}\right)\right)\right]$$

Probabilistic model

- **Assumption**: measurements come from a computational model + noise

$$\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{r}; \boldsymbol{s}) + \varepsilon$$

- **Assumption:** noise is normally distributed

$$\varepsilon \sim \mathcal{N}(0, \Sigma(\boldsymbol{s}))$$

- **Assumption**: correlations

$$\Sigma_{ij}(\boldsymbol{s}) = \begin{cases} \sigma^2 \exp\left(-\dfrac{\left\| \boldsymbol{x}(s_i) - \boldsymbol{x}(s_j) \right\|}{\ell} \right), & \text{if } 1 \le i, j \le n \\ \Sigma_{i-n,j-n}(\boldsymbol{s}), & \text{if } n < i, j \le 2n \\ 0 & \text{otherwise} \end{cases} .$$

Papadimitriou, et al., 2012

Optimization

- Optimization in a high dimensional space (number of sensors):

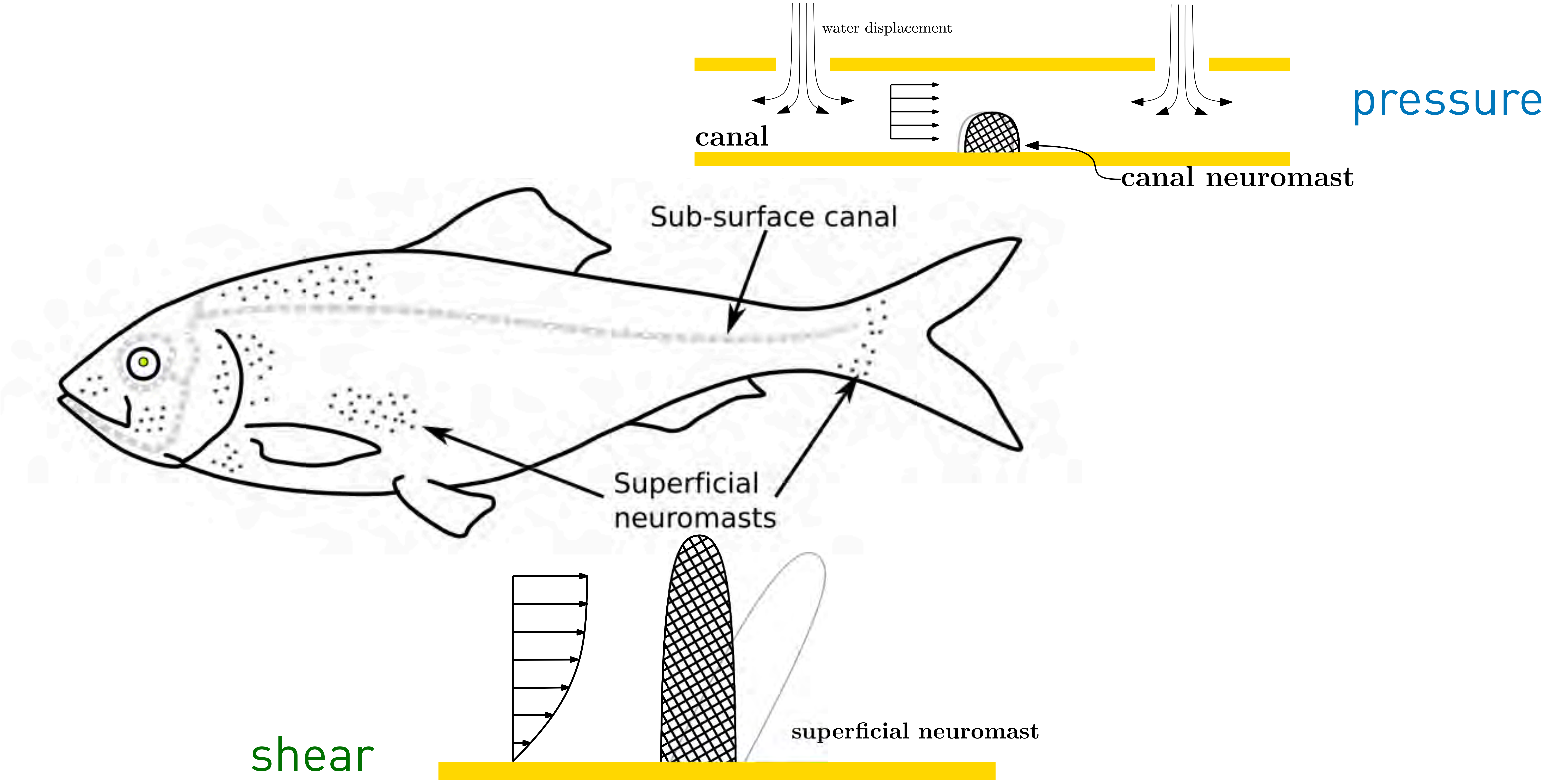
$$s^\star = \underset{s}{\operatorname{argmax}}\, \hat{U}(s)$$

- **Sequential optimization** *(Papadimitriou C, J. Sound. Vibr., 2004)*

$$\hat{U}_n(s) = \hat{U}\left(s_1^\star, \ldots, s_{n-1}^\star, s\right)$$

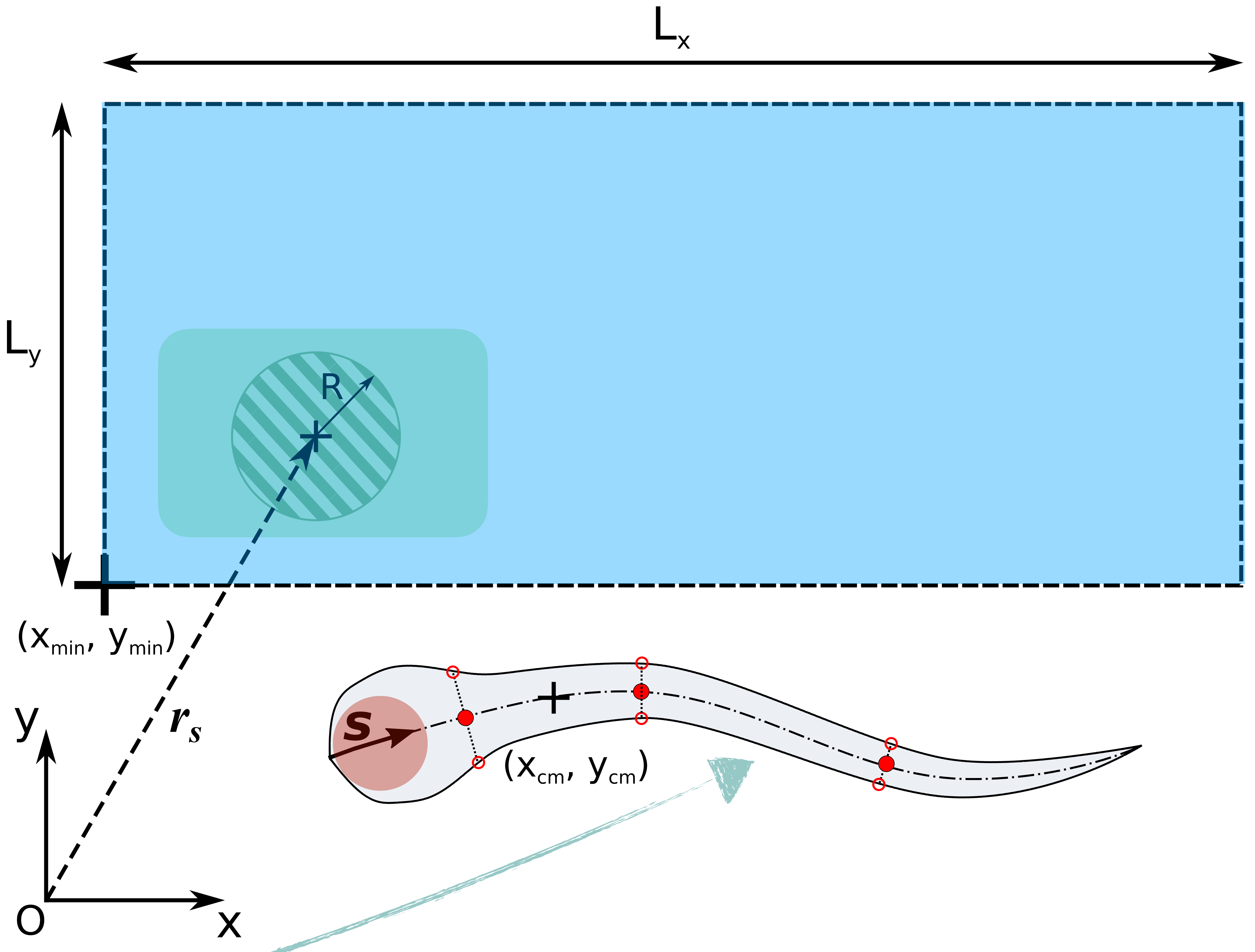
$$s_n^\star = \underset{s}{\operatorname{argmax}}\, \hat{U}_n(s) \quad \text{and} \quad \hat{U}_n^\star = \max_{s} \hat{U}_n(s)$$

Lateral Line Organ - Neuromasts



Simulation setup

Location of the disturbance
$\boldsymbol{r}$



Prior belief of the disturbance
$p(\boldsymbol{r})$

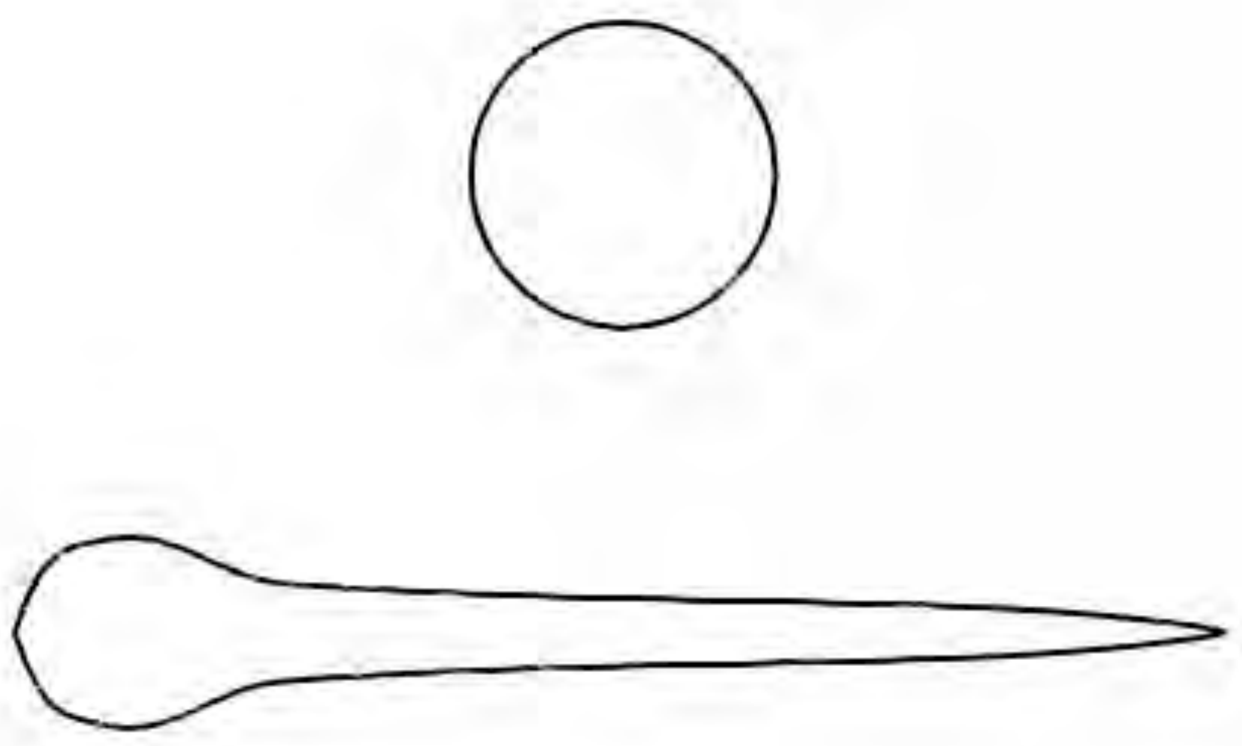
Measurements on the skin
$\boldsymbol{y}$

Sensor Location
$\boldsymbol{s}$

By Solving

$$\frac{\partial \omega}{\partial t} + (\boldsymbol{u} \cdot \nabla)\omega = \nu \nabla^2 \omega + \lambda \nabla \times \left(\chi \left(\boldsymbol{u}_s - \boldsymbol{u} \right) \right)$$

1. Static Swimmer

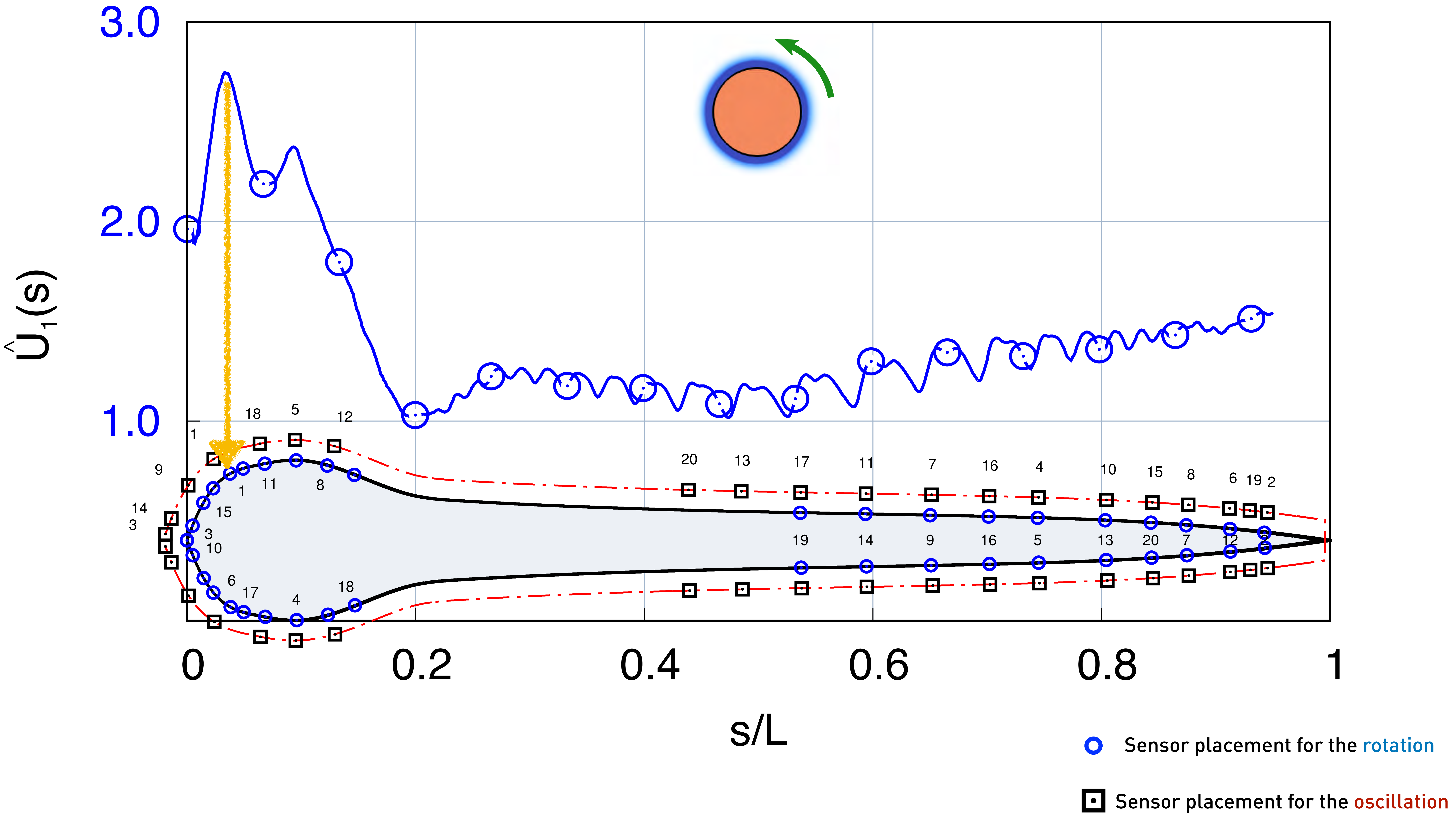


- stationary larva-shaped Zebra fish (Engert Lab, Harvard University) $L = 0.5$
- cylinder with diameter $D = 0.25L$ that
  1. oscillates parallel to the 'anteroposterior' axis of the body $\omega_{\text{osz}} = 10\,\text{Hz}$
  2. rotates with constant angular velocity $\omega_{\text{rot}} = 10\,\text{Hz}$
- kinematic viscosity $\nu = 10^{-4}$
- the disturbance-sources start moving at 0s, and time-averaging of the recorded data is done between 0.95s and 1.0s

1. Static Swimmer

$$U(\boldsymbol{s}) = \int_{\mathcal{Y}} \int_{\mathcal{R}} \log \frac{p(\boldsymbol{y}|\boldsymbol{r},\boldsymbol{s})}{p(\boldsymbol{y}|\boldsymbol{s})}\, p(\boldsymbol{r})\, p(\boldsymbol{y}|\boldsymbol{r},\boldsymbol{s})\, \mathrm{d}\boldsymbol{r}\, \mathrm{d}\boldsymbol{y}$$

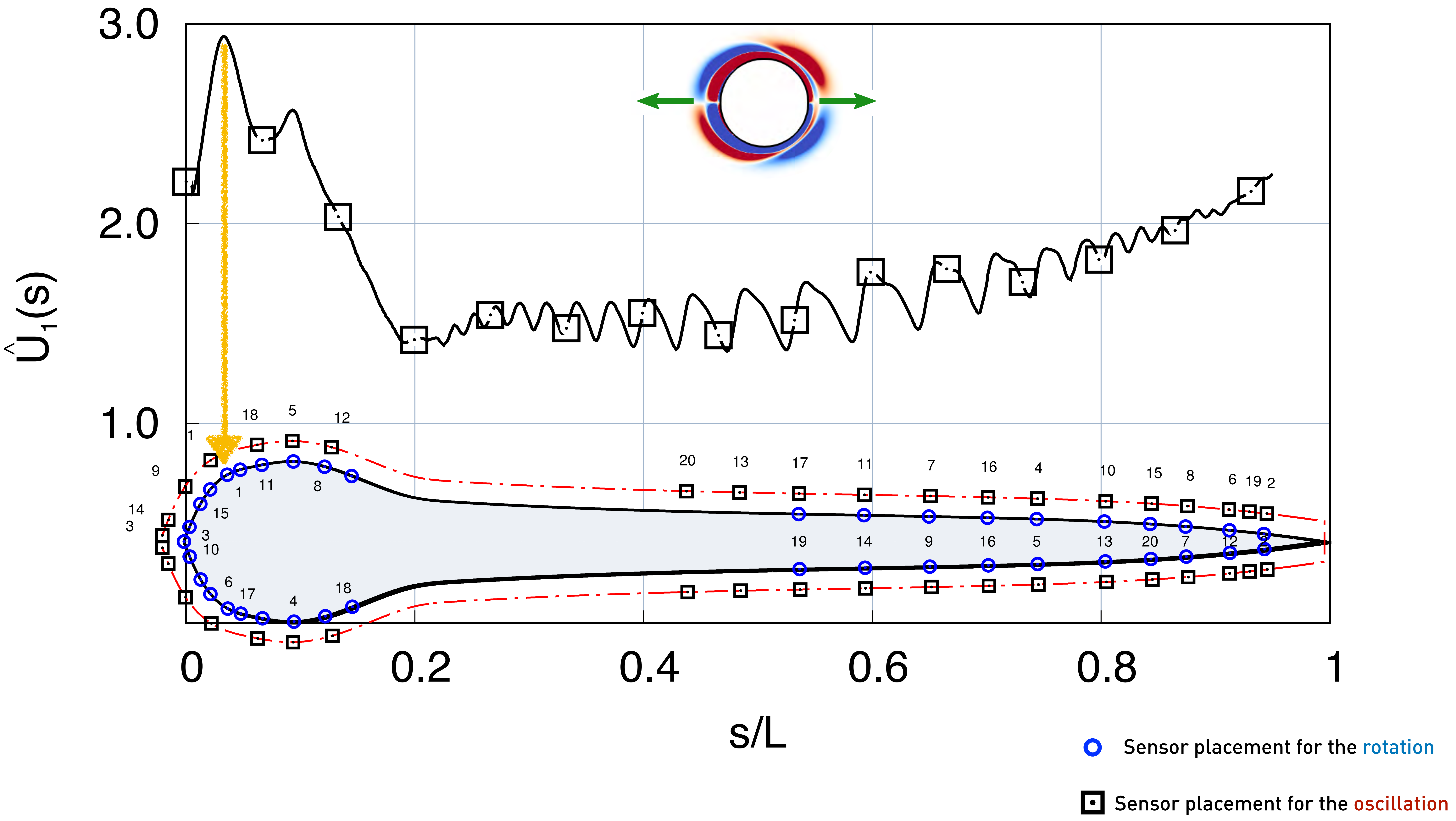
Utility function for rotation and one sensor



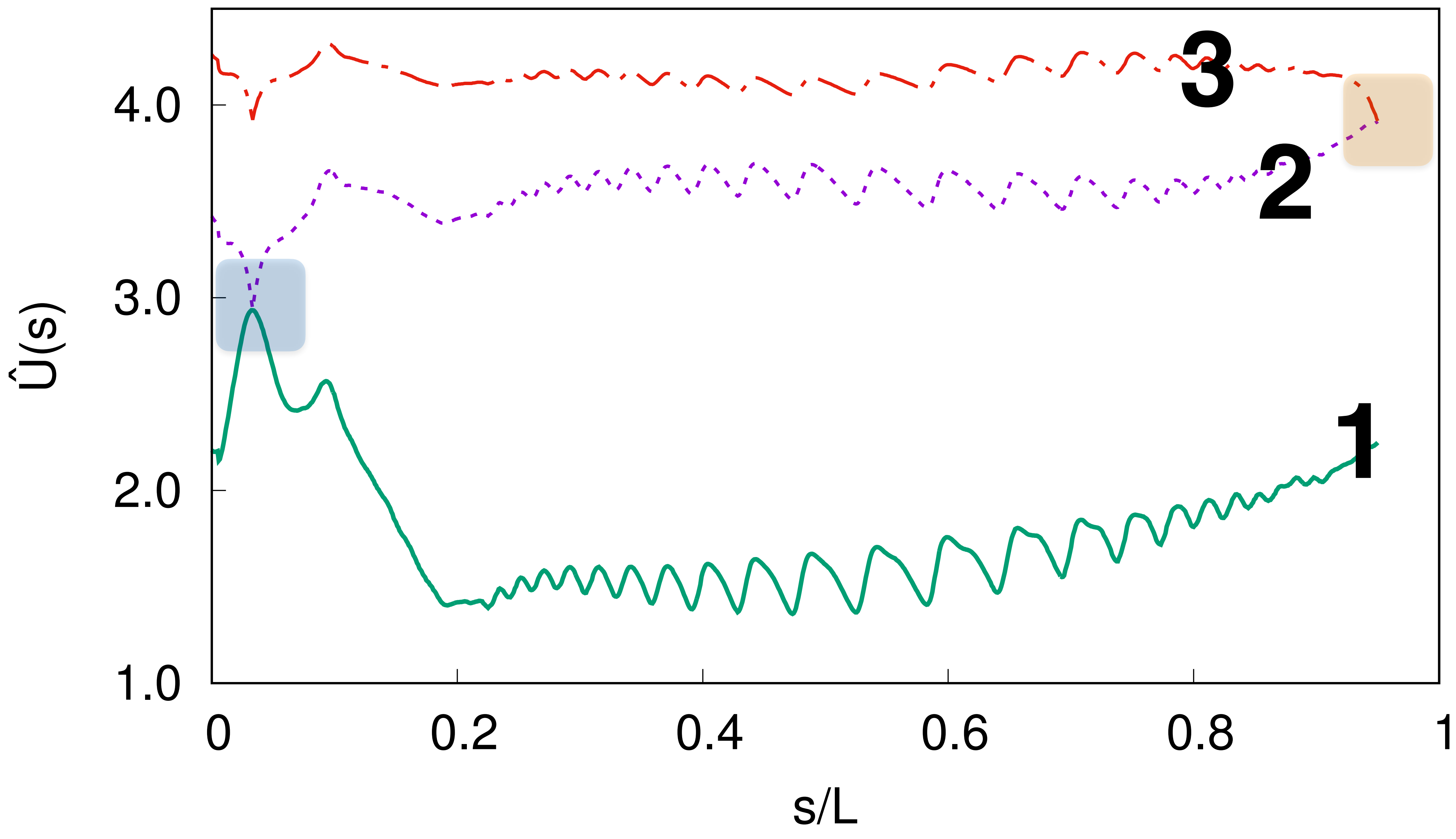
1. Static Swimmer

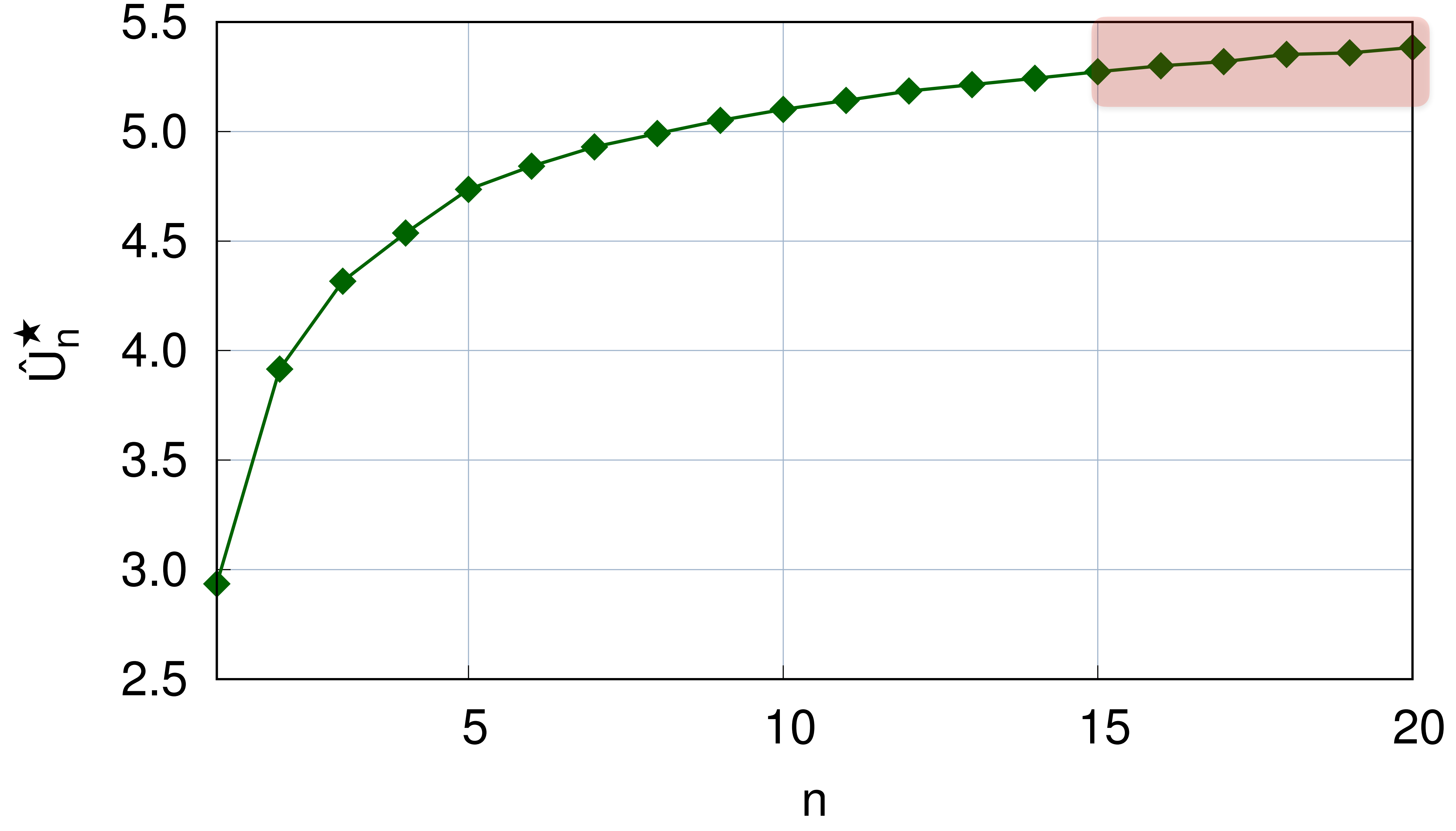
$$U(\boldsymbol{s}) = \int_{\mathcal{Y}} \int_{\mathcal{R}} \log \frac{p(\boldsymbol{y}|\boldsymbol{r}, \boldsymbol{s})}{p(\boldsymbol{y}|\boldsymbol{s})} \, p(\boldsymbol{r}) \, p(\boldsymbol{y}|\boldsymbol{r}, \boldsymbol{s}) \, \mathrm{d}\boldsymbol{r} \, \mathrm{d}\boldsymbol{y}$$

Utility function for oscillation and one sensor



1. Static Swimmer



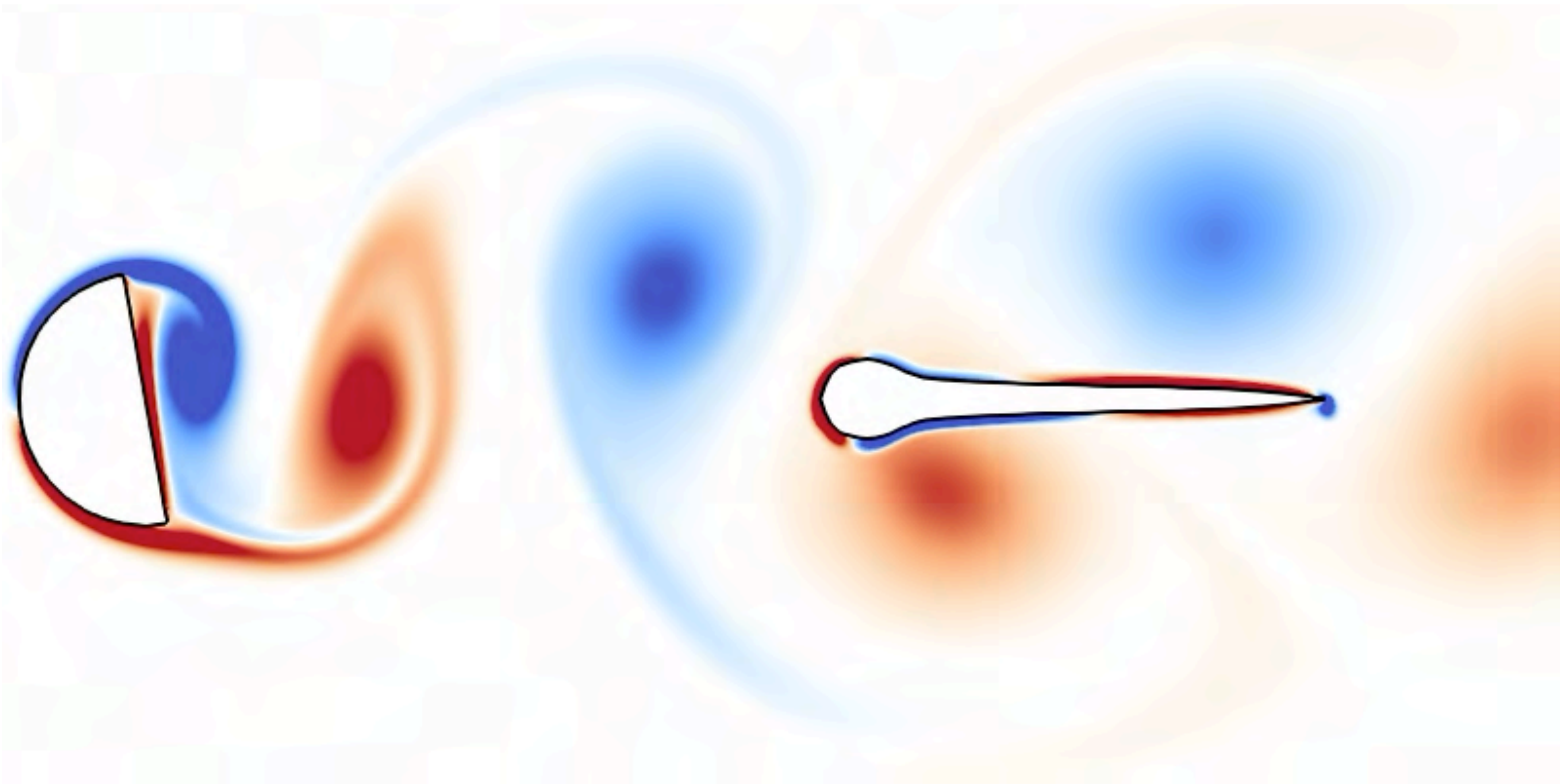


The utility of sensor 2 is minimum at the maximum of the sensor 1

The utility of sensor 3 is minimum at the maximum of the sensor 2

The maximum utility stops changing as we add more sensors

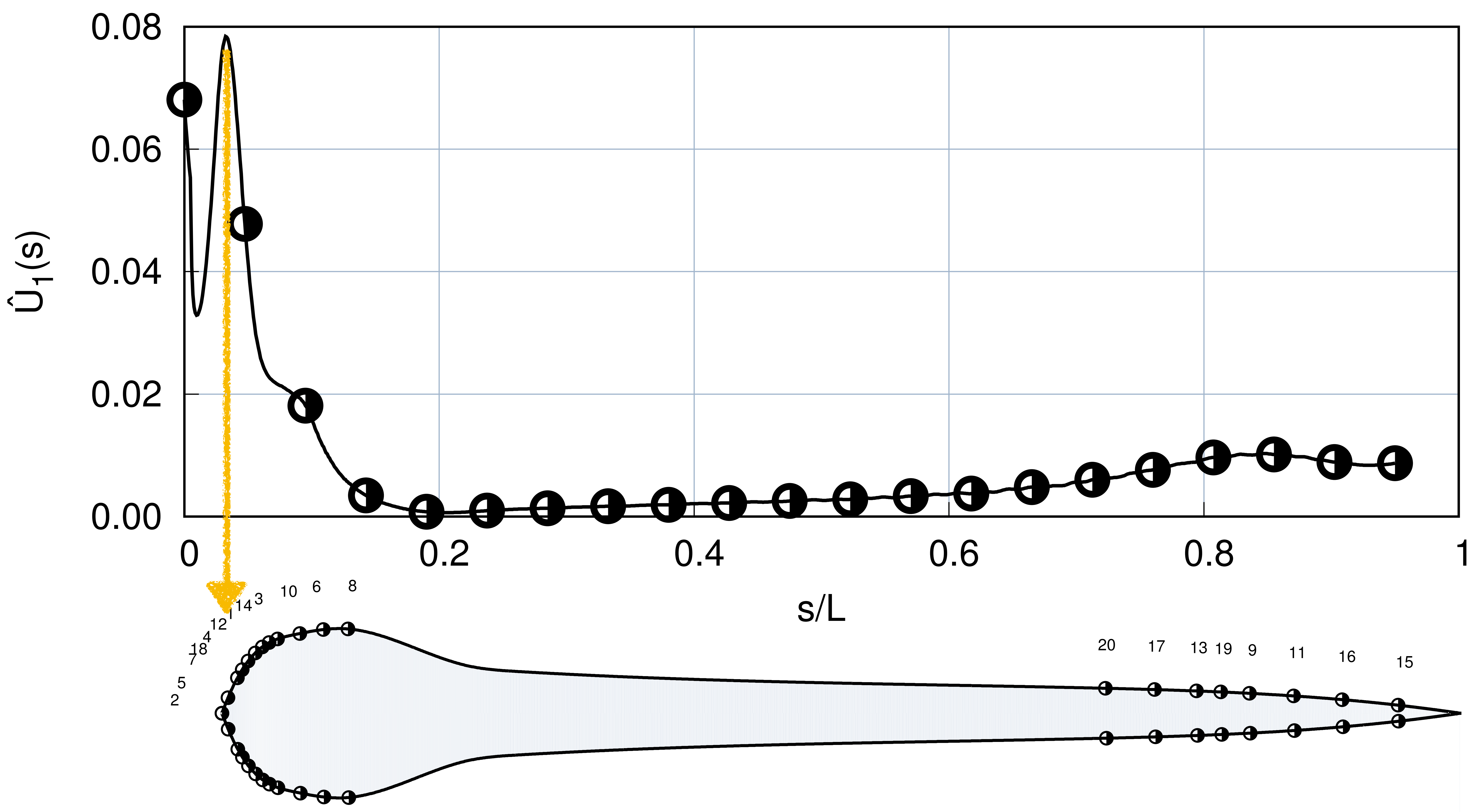
2. Static swimmer in the wake of a D-cylinder



- rigid larva-shaped profile
- placed in the unsteady vortex-wake generated by a D-shaped half cylinder
- Re=200 based on the cylinder diameter, and $Re=400$ based on the swimmer length
- time-averaging of the recorded data is done between 0.95s and 1.0s
- shear stress measurements

2. Static swimmer in the wake of a D-cylinder

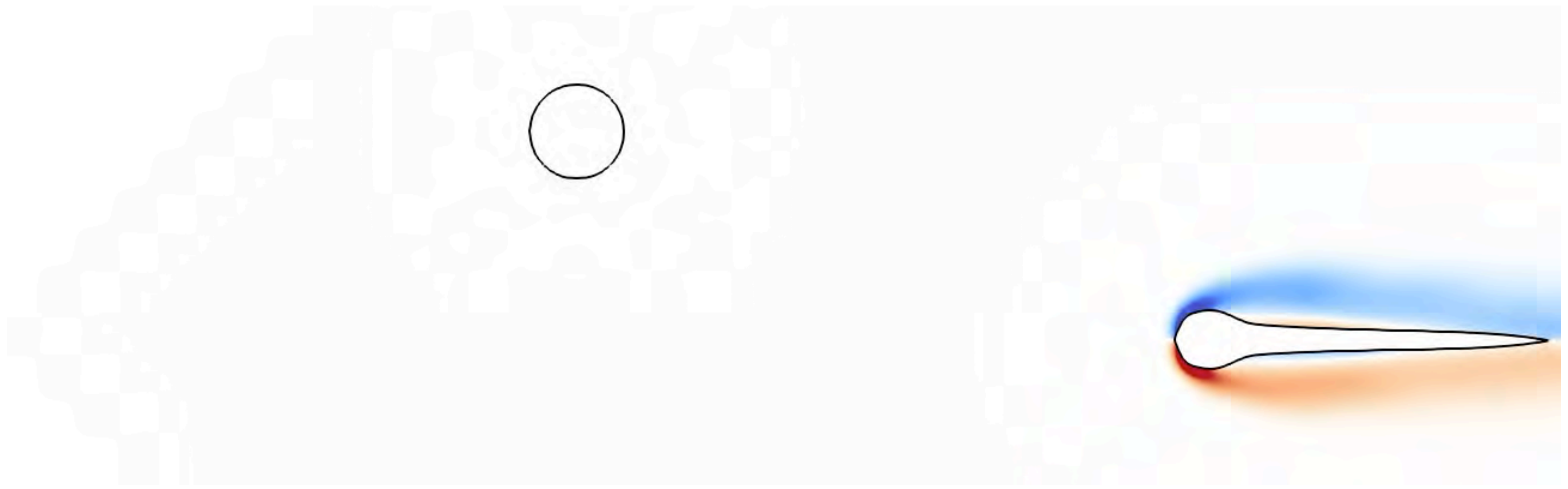
Utility function for D-shaped cylinder and one sensor



3. Self-propelled swimmers

- $\mathrm{Re} \approx 280$

Larva swimmer/Rotating Cylinder

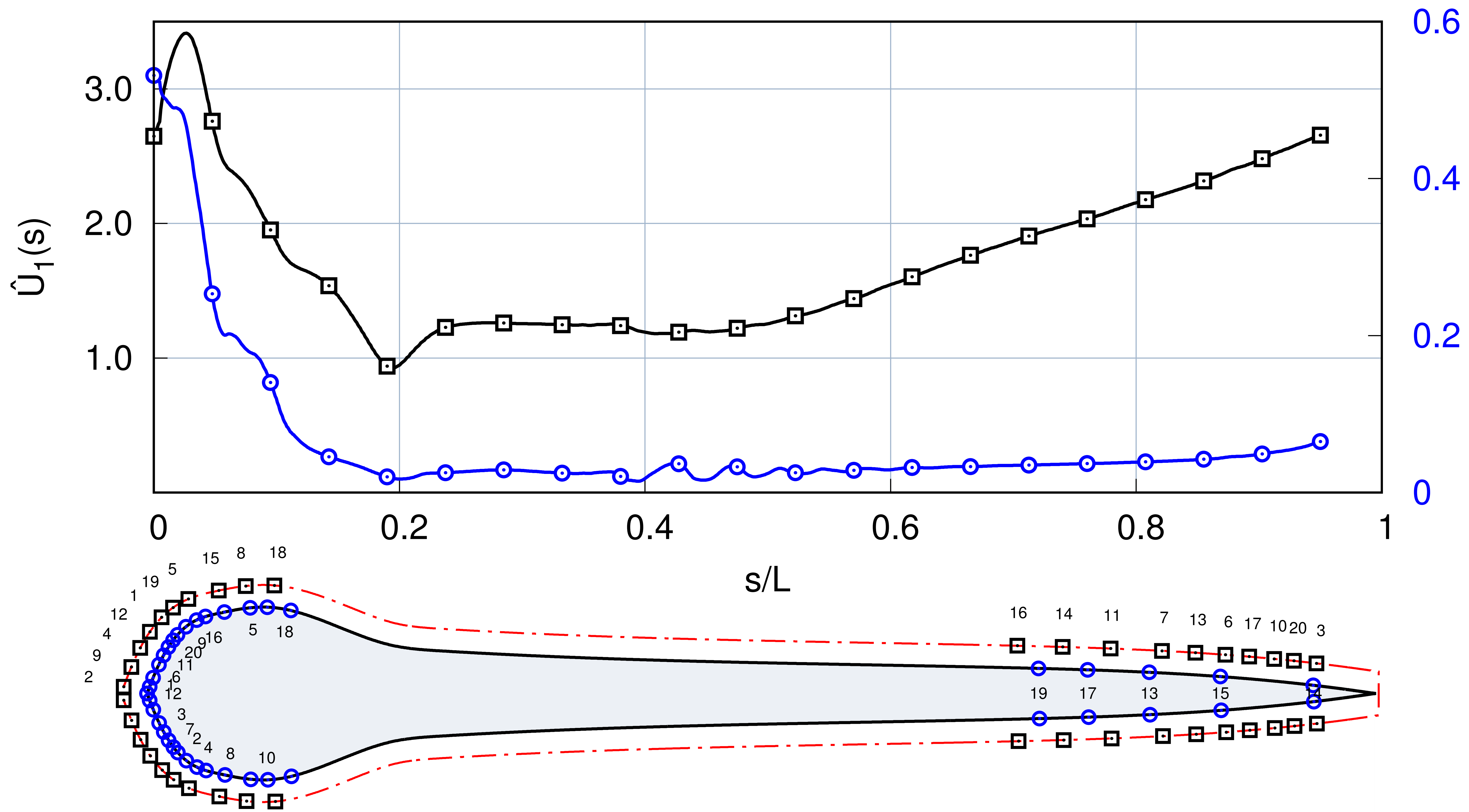


“Adult” swimmer/Oscillating Cylinder

- swimmers perform four full burst-coast swimming cycles starting from rest, before the cylinder starts rotating/oscillating.
- shear stress measurements taken towards the end of a coasting phase to allow self-generated disturbances to subside sufficiently, and are averaged over a small time window

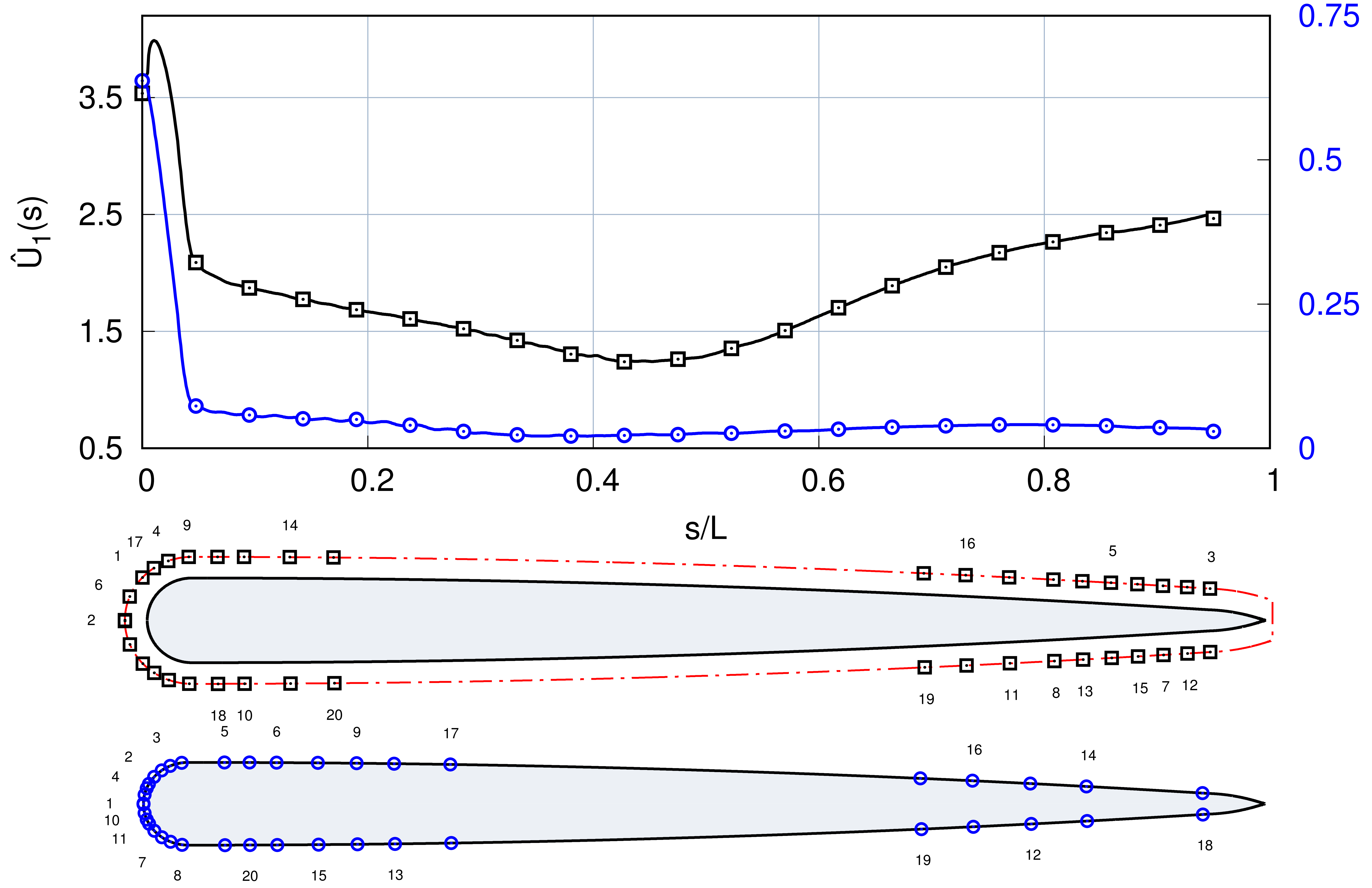
3. Self-propelled swimmers: larva

Utility function for self propelled larva and one sensor



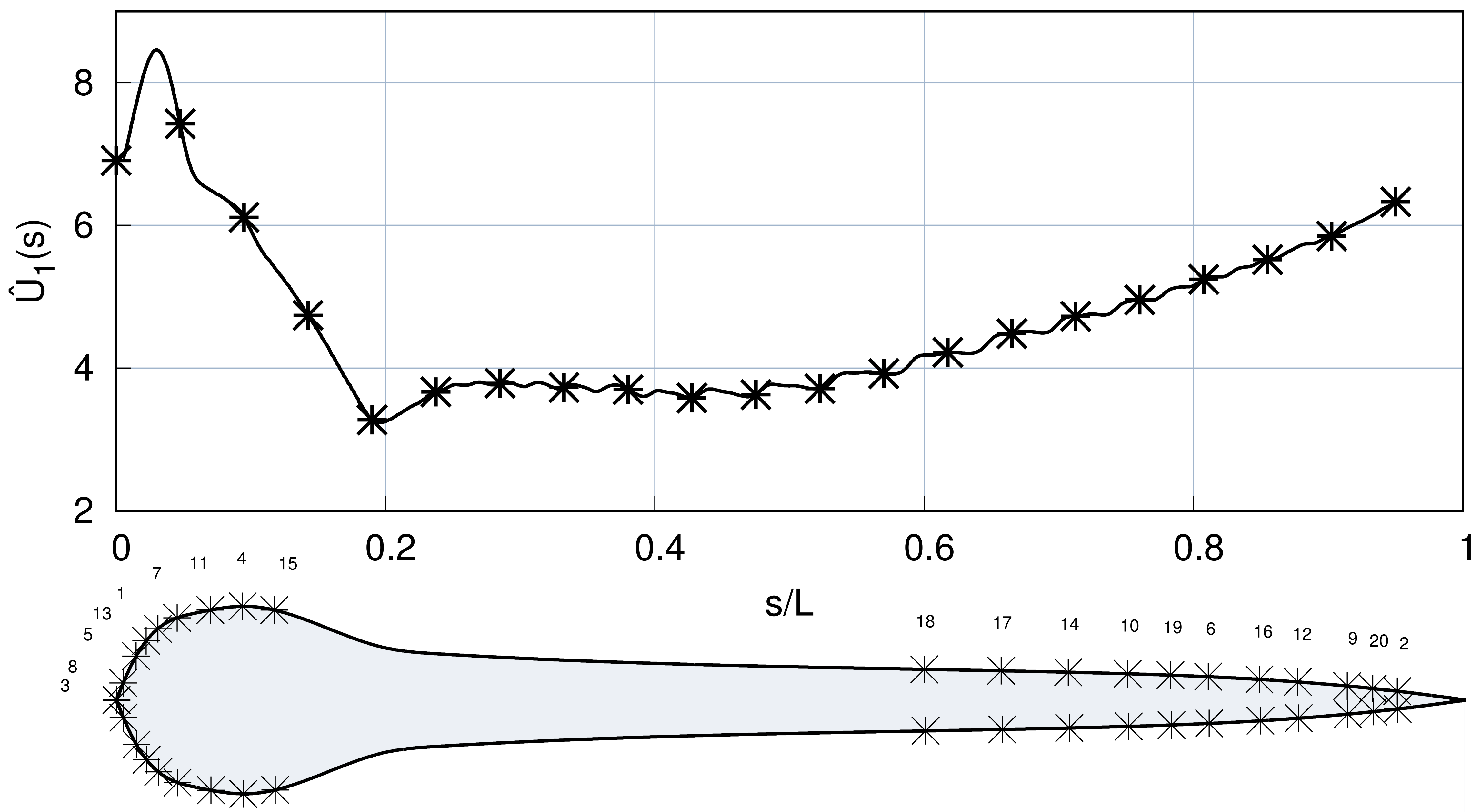
3. Self-propelled swimmer - AIRFOIL SHAPE

Utility function for self propelled airfoil and one sensor

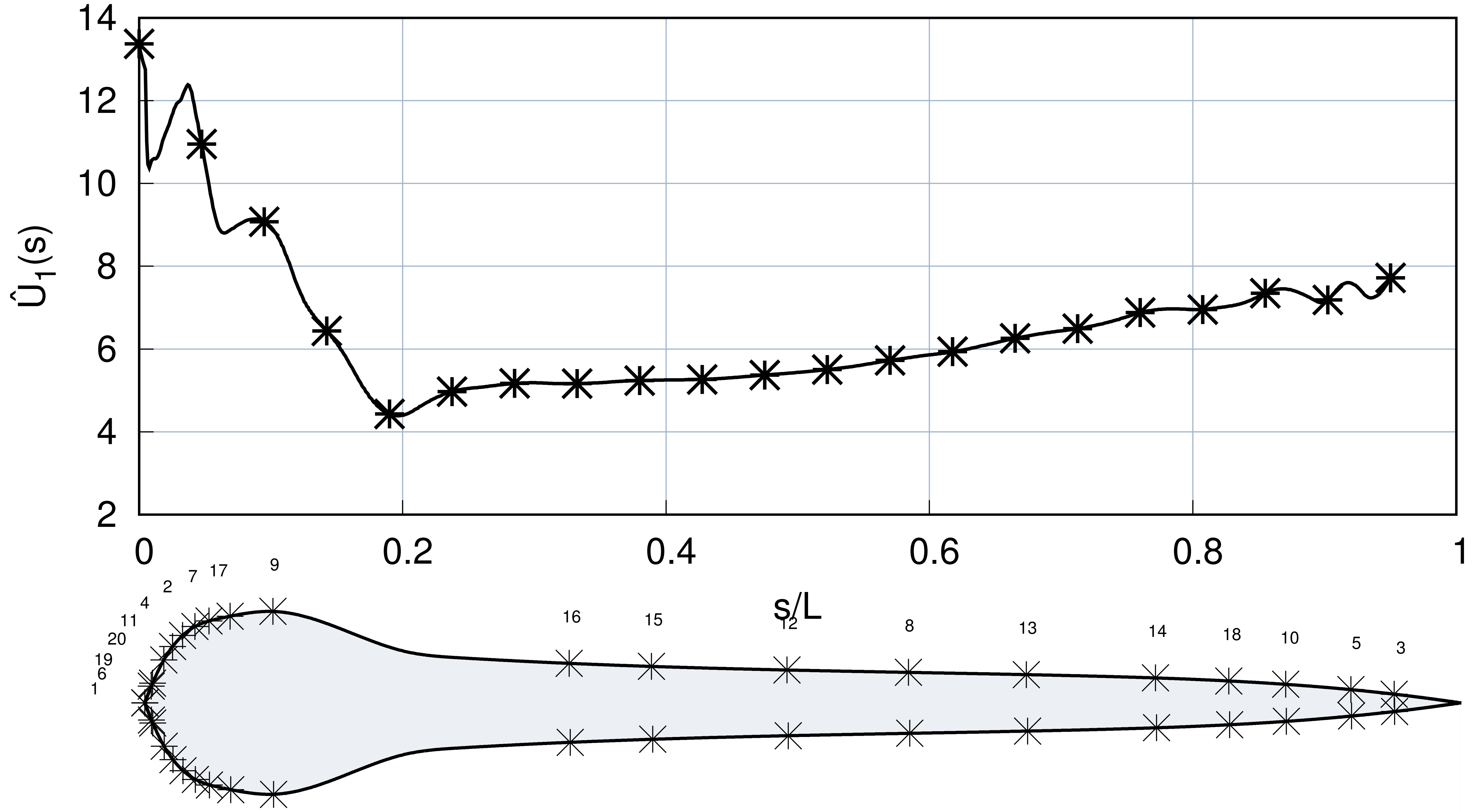


Combination of all experiments

Shear stress

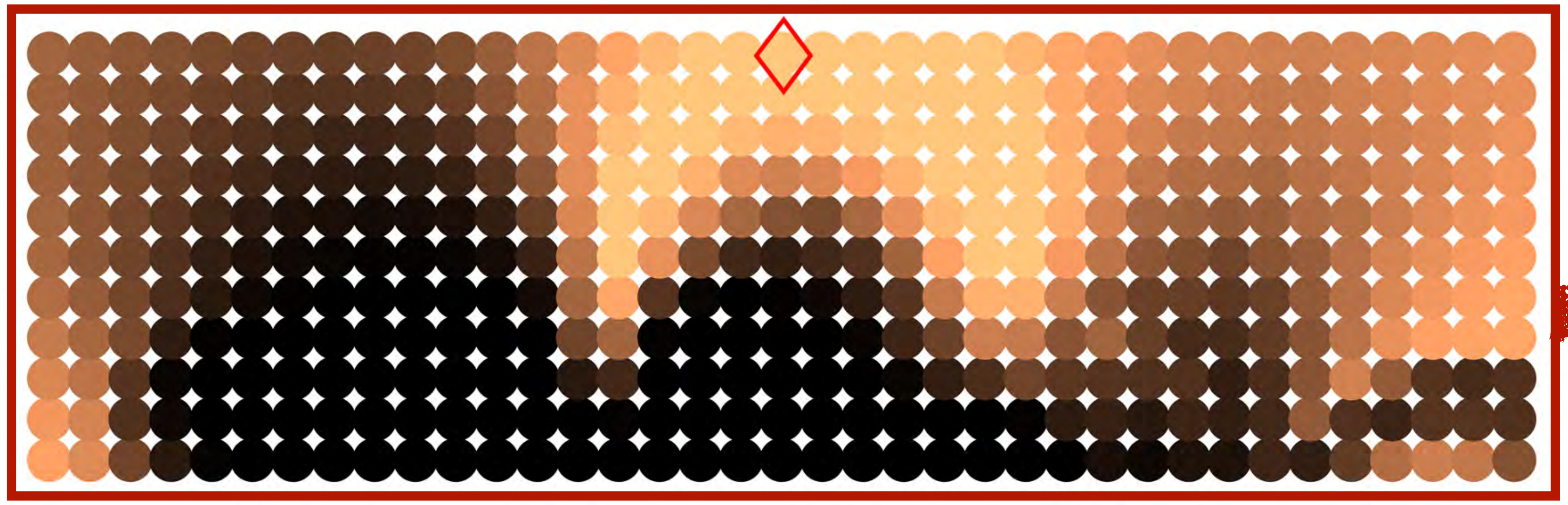


Pressure gradient

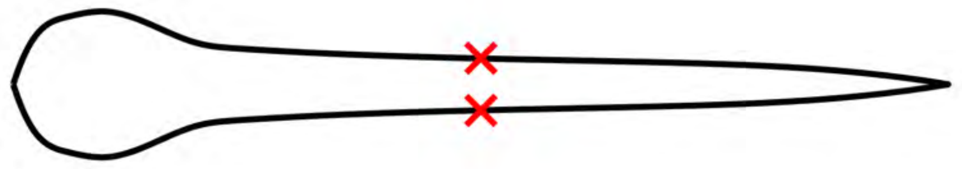


Inference of the disturbance location

Disturbance actual position



Prior belief of the location of the disturbance



x 1 Pair of sensors positioned in the middle of the fish

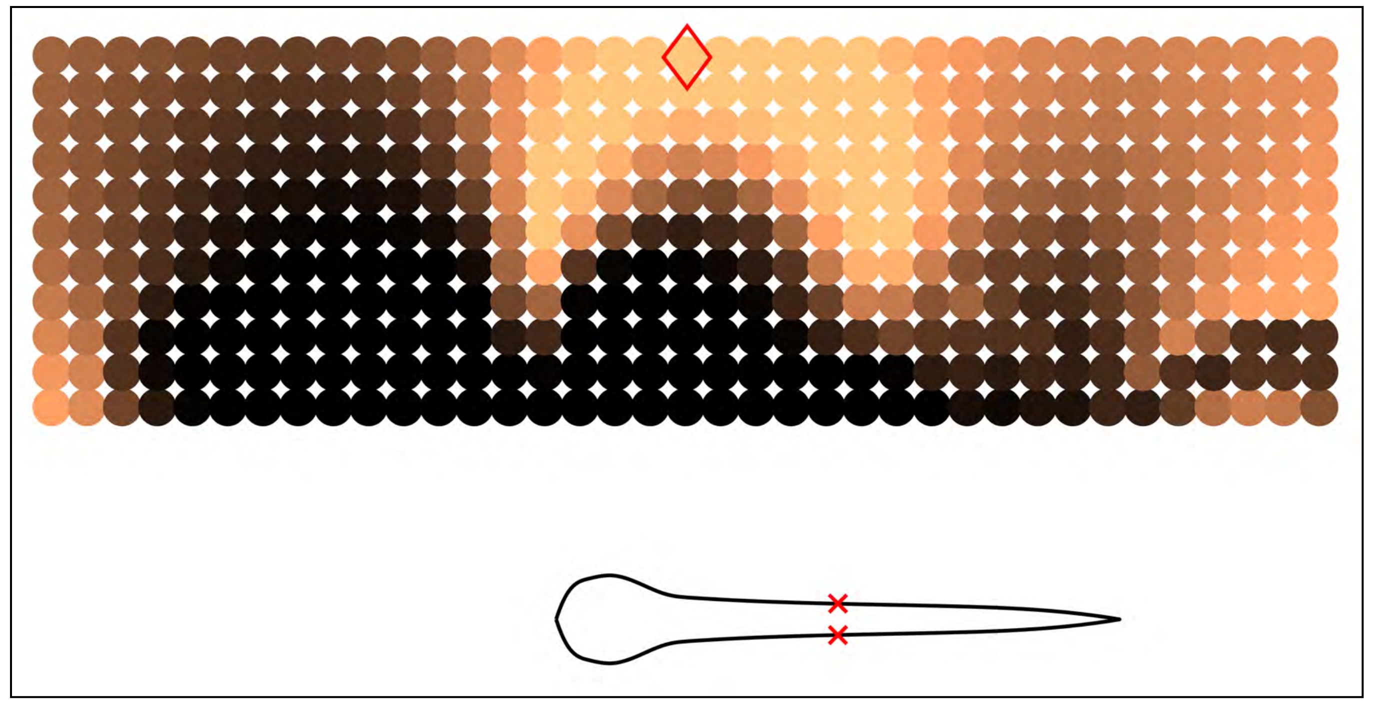
The colours correspond to the value of the posterior belief of the location of the disturbance

$$p(\boldsymbol{r}|\boldsymbol{y},\boldsymbol{s})$$

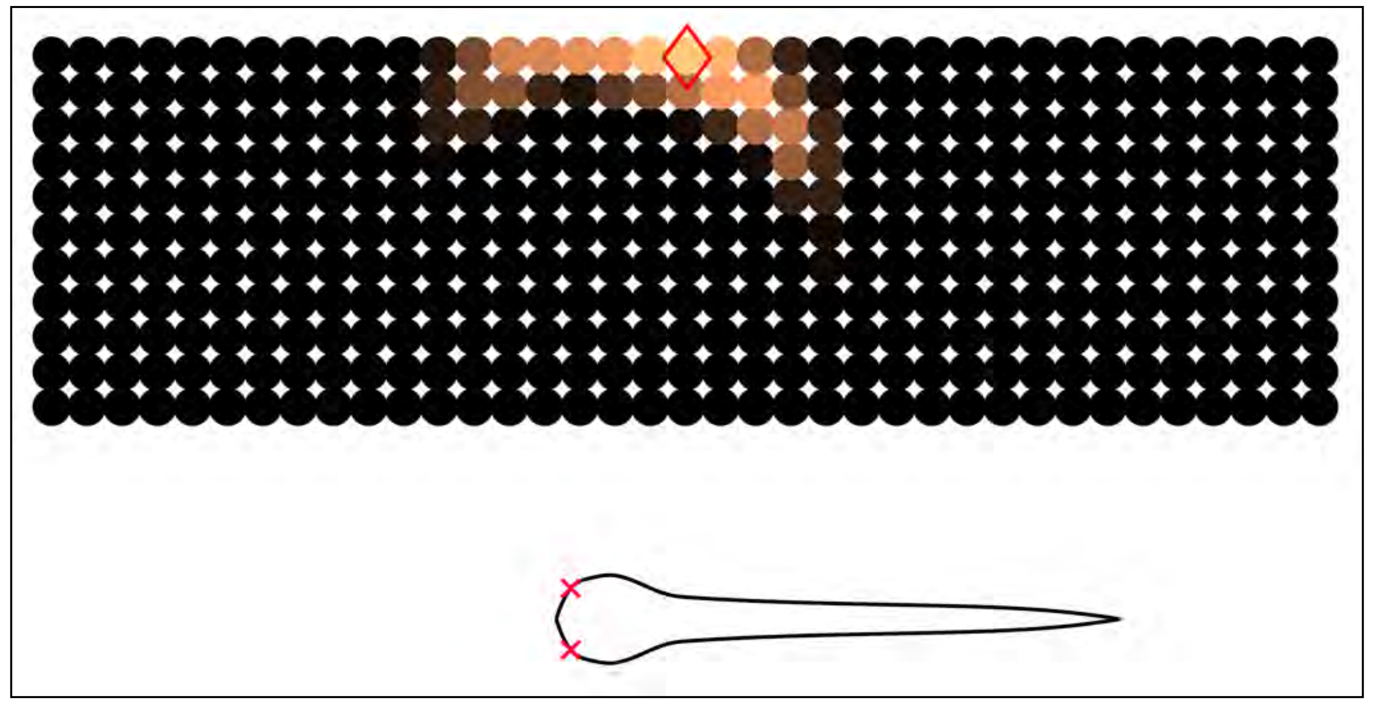
Dark colours correspond to low values and light to high values

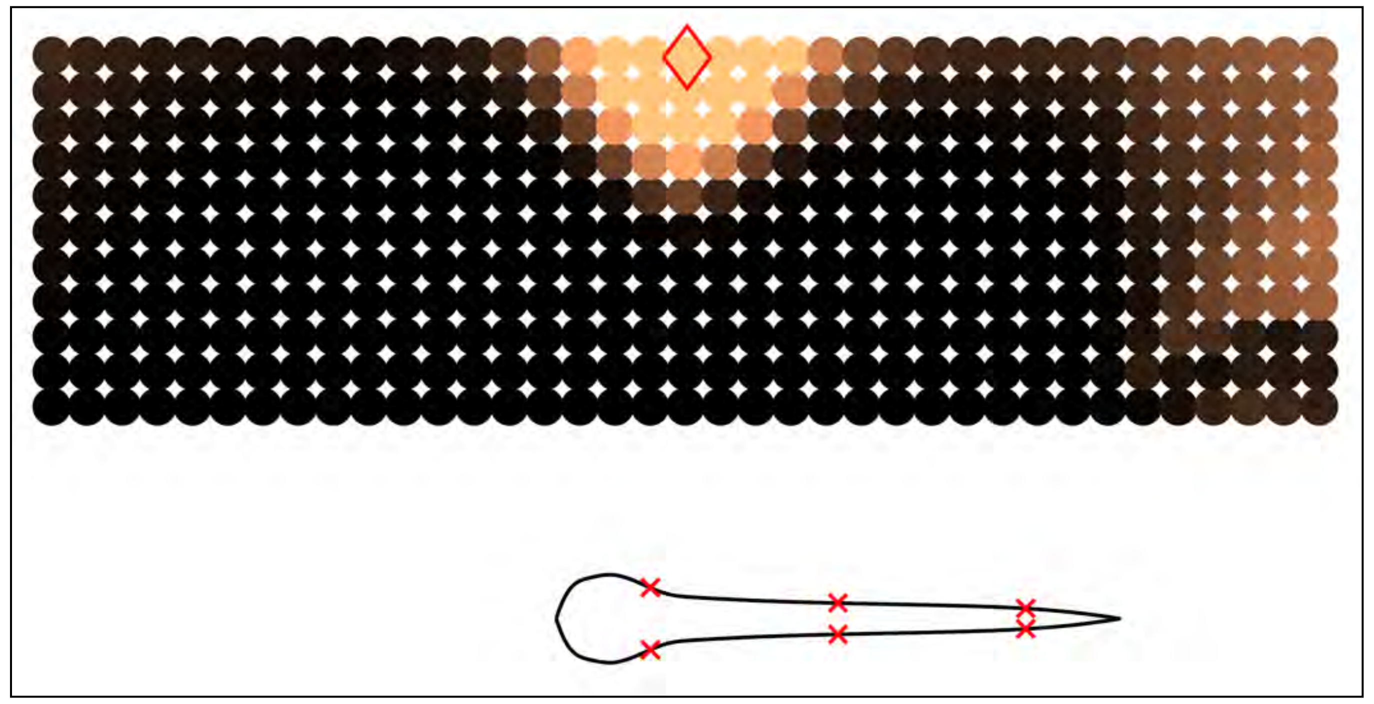
Inference of the disturbance location

uniform sensors | **OPTIMAL sensors**

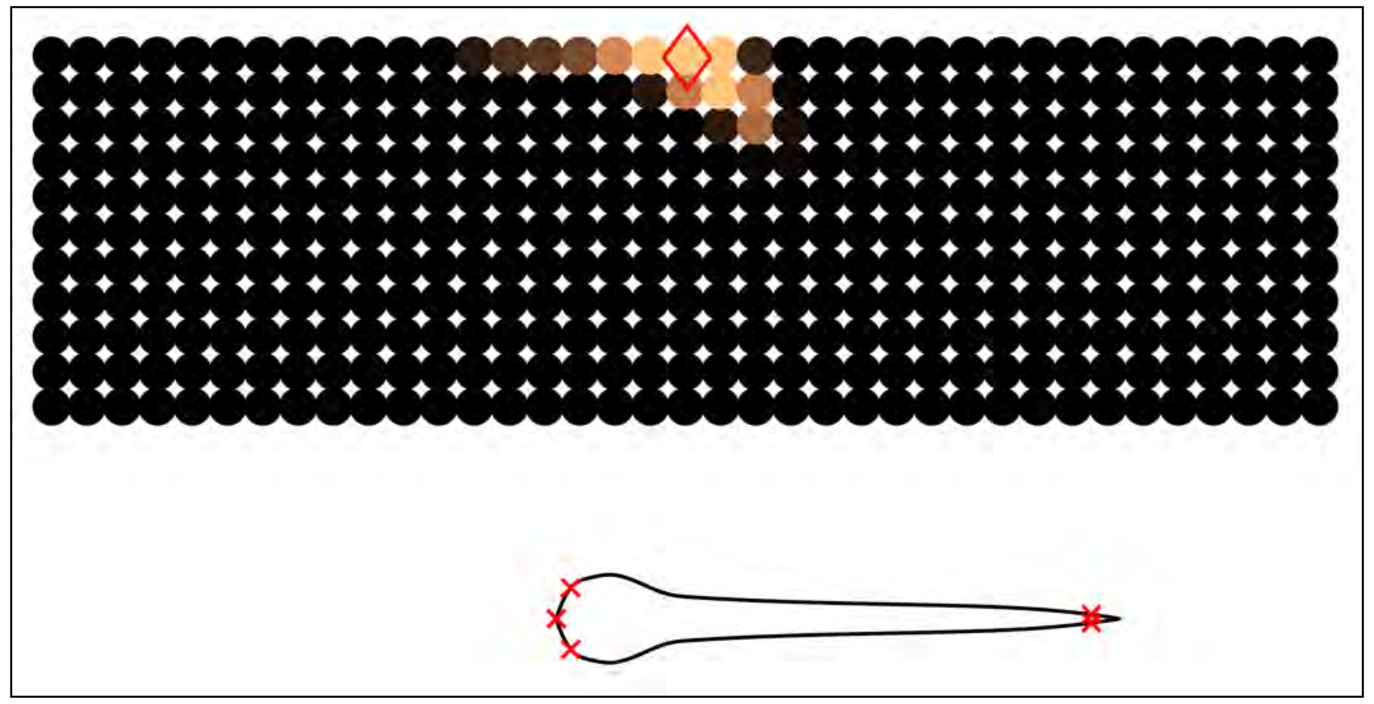


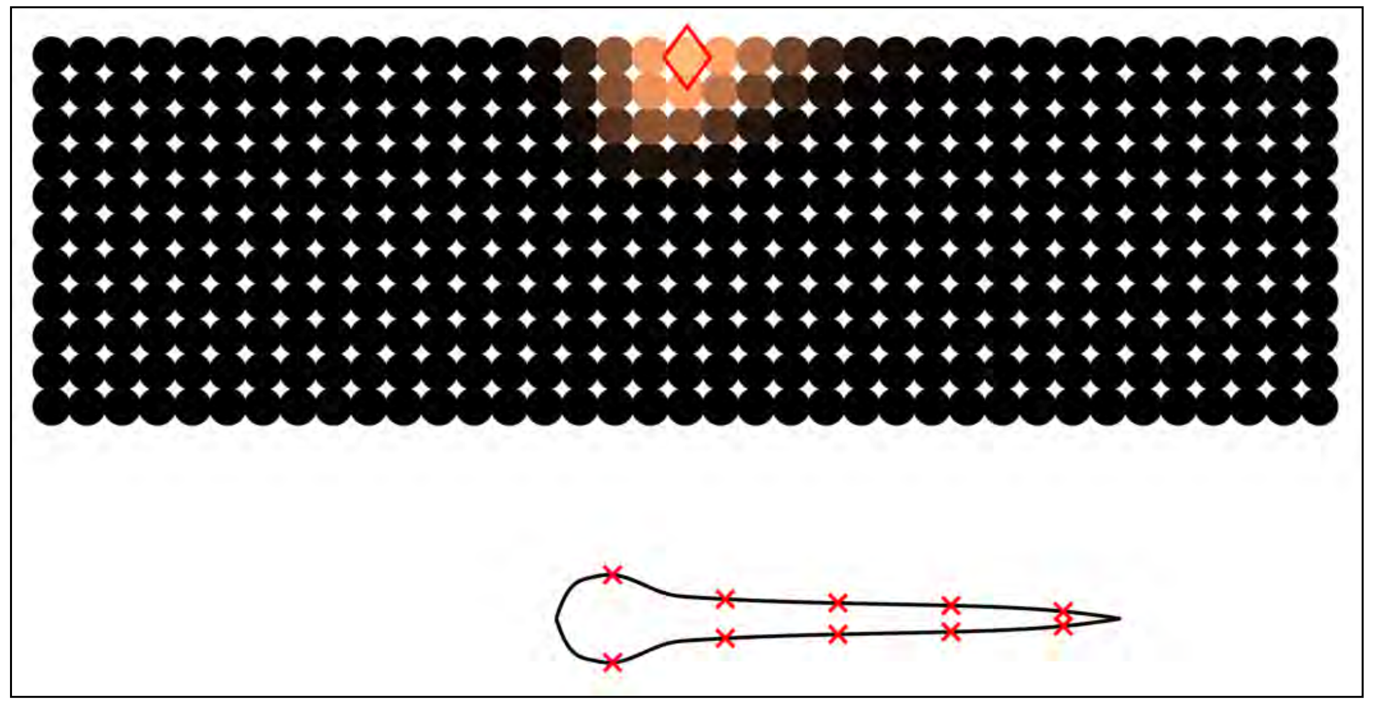
2 sensors



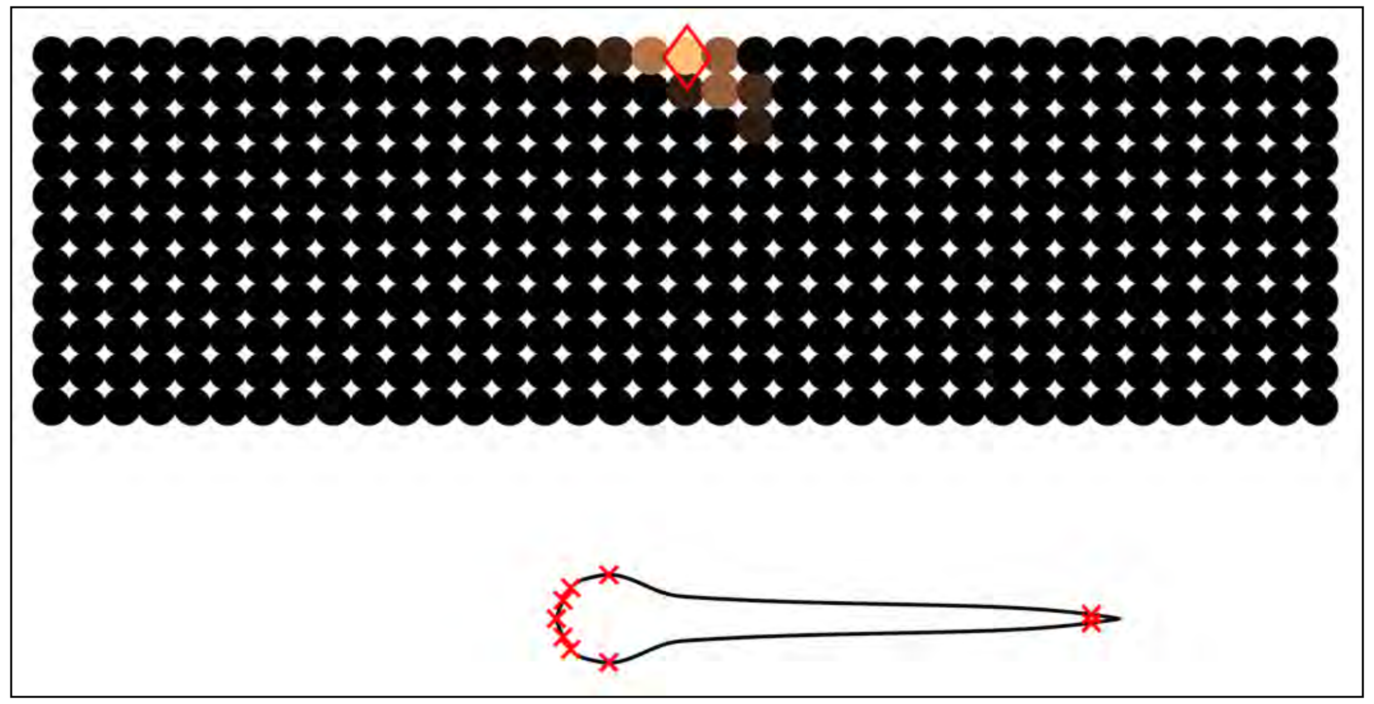


6 sensors





10 sensors



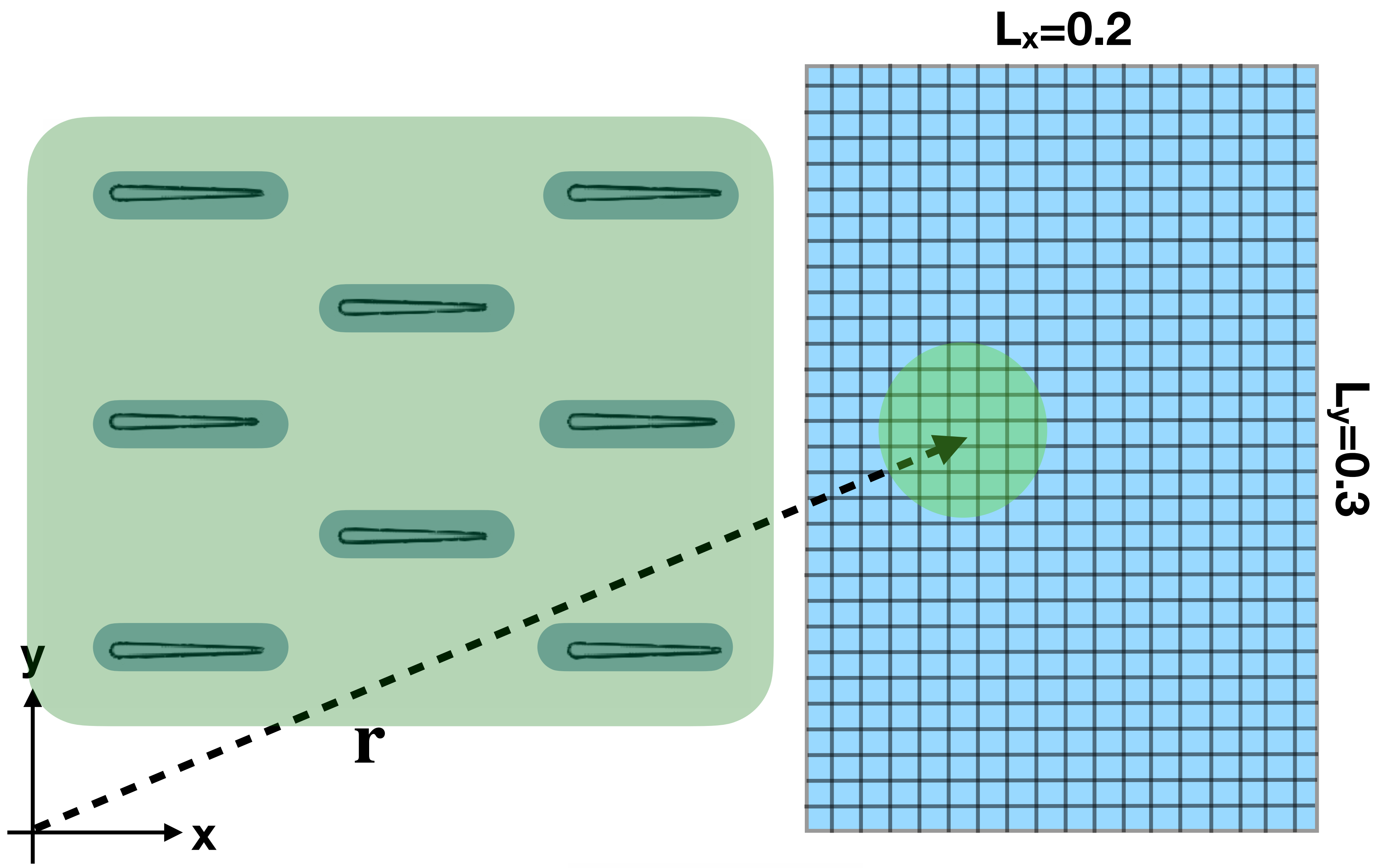
Optimal Experimental Design:

Optimal sensor placement for fish shoals

Simulation setup

Initial location
$\mathbf{r}$

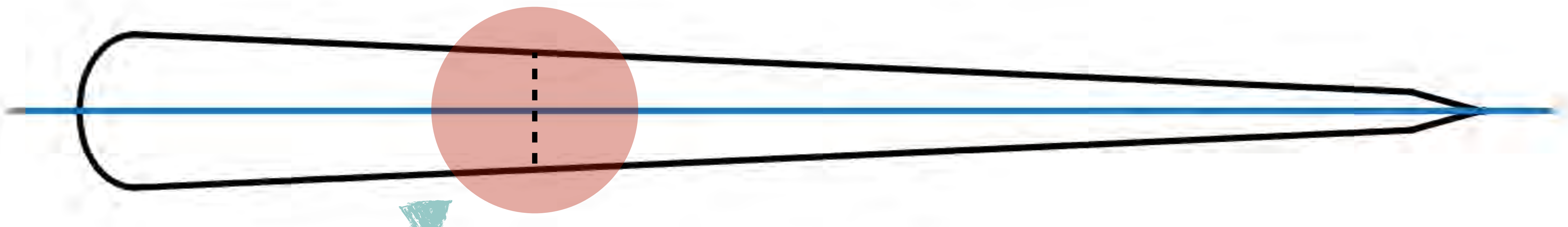
Number of fish in leading group
$\boldsymbol{\Theta}$



Prior belief of the disturbance
$p(\mathbf{r})$

$p(\boldsymbol{\Theta})$

Measurements on the skin
$\boldsymbol{y}$



Sensor Location
$\boldsymbol{s}$

By Solving

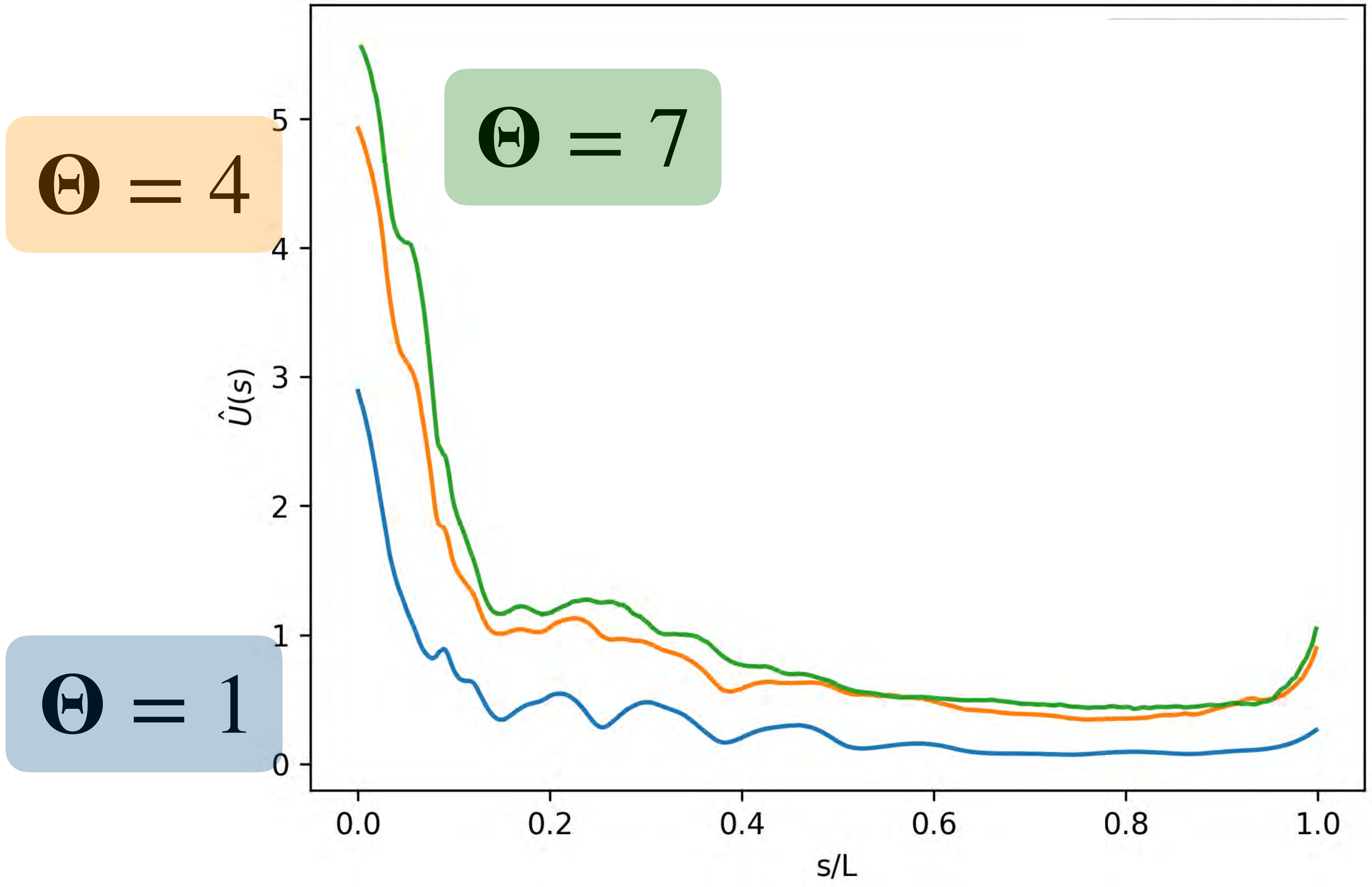
$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p/\rho + \nu \Delta \mathbf{u} + \lambda \chi (\mathbf{u}_s - \mathbf{u})$$

$$\nabla \cdot \mathbf{u} = 0$$

Varying size of leading group

Varying size of leading group

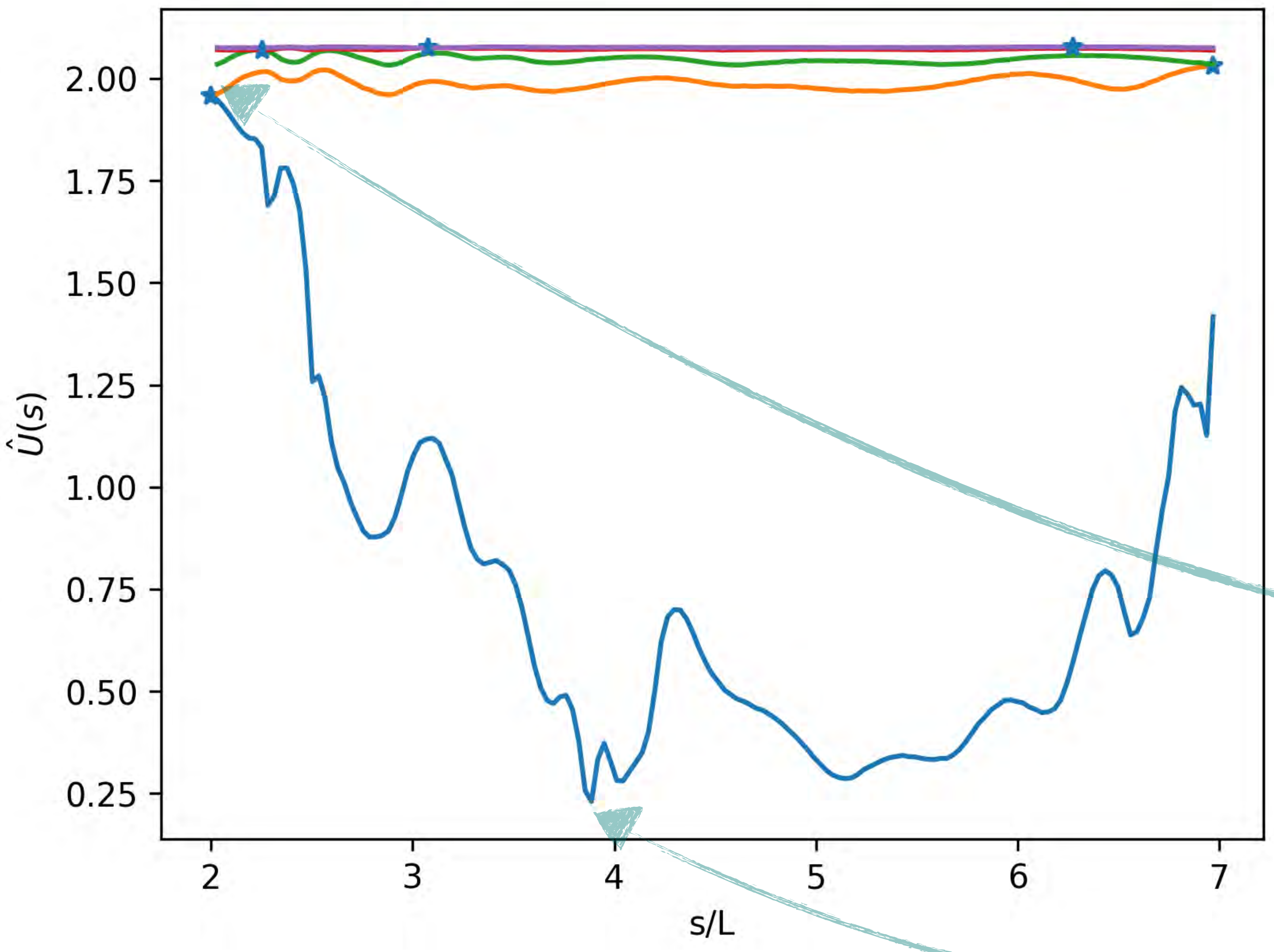
Utility function to infer position of leading fish

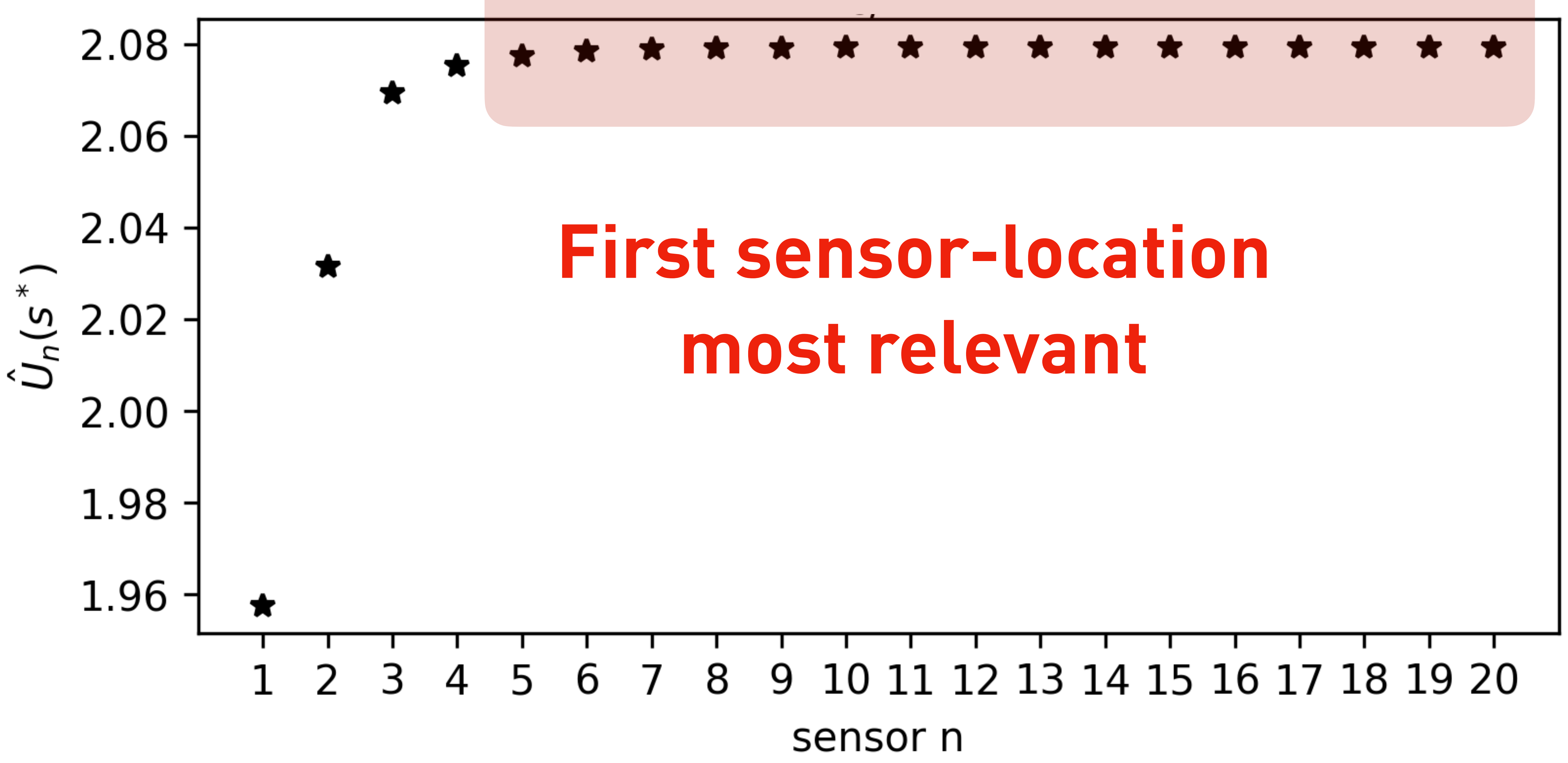


The utility for $\mathbf{r}$ is not influenced by number of fish

Inferring size of the leading group

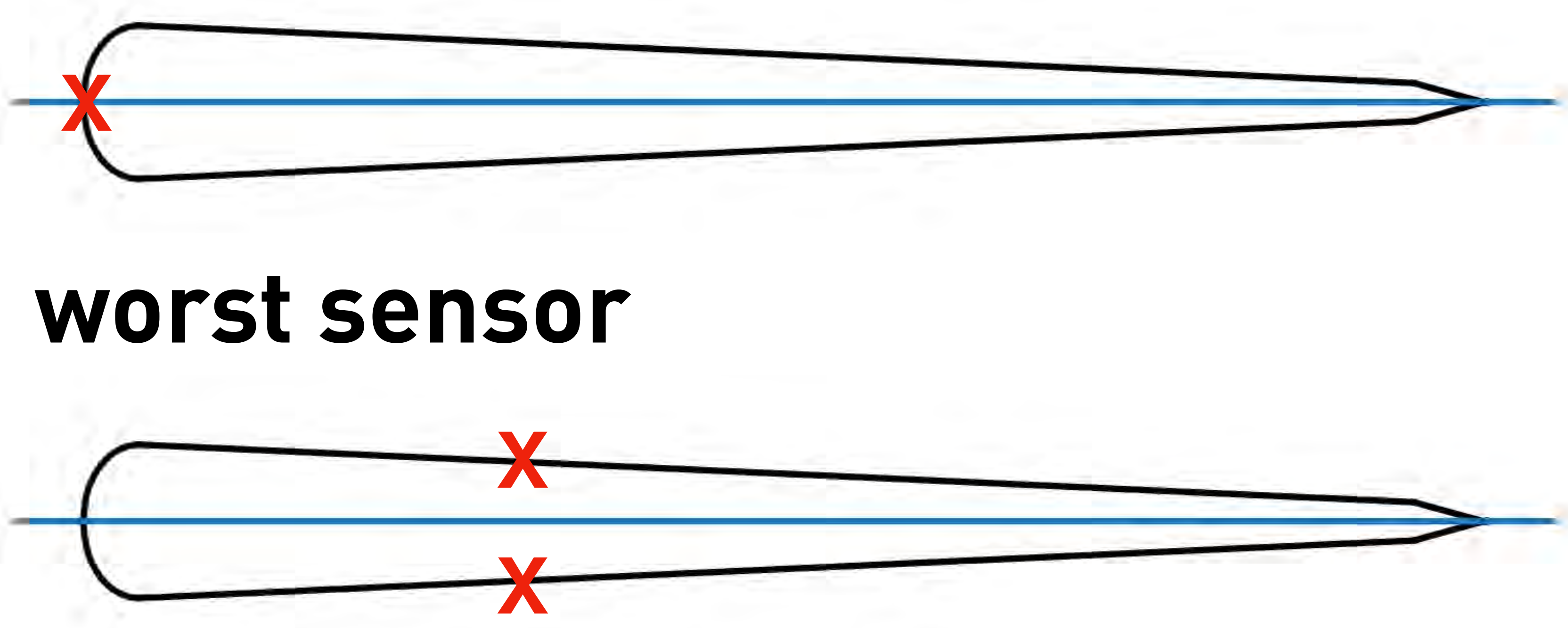
Utility function for first sensor to infer number of leading fish





best sensor

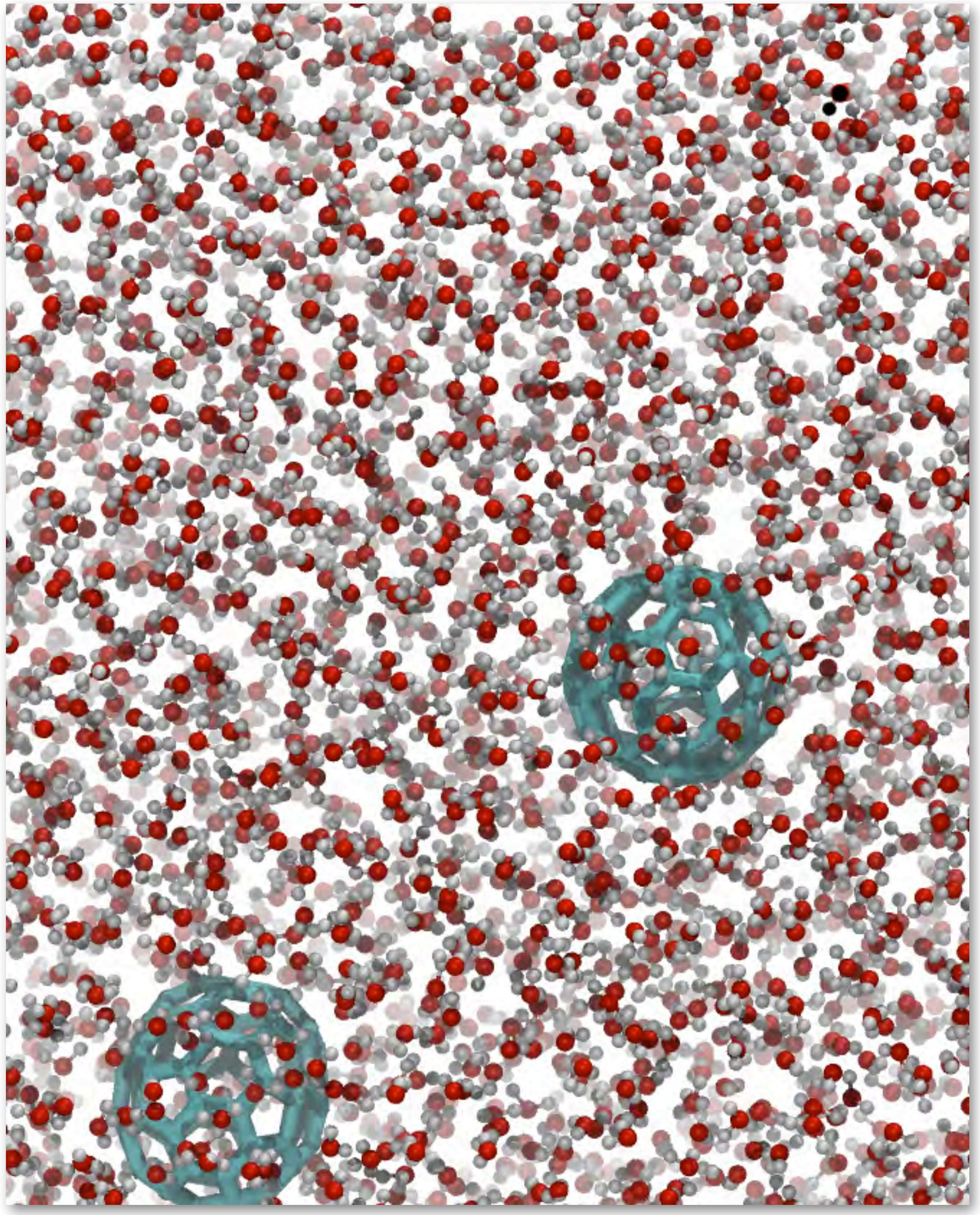
worst sensor



LEARNING, MACHINES AND FLUID MECHANICS

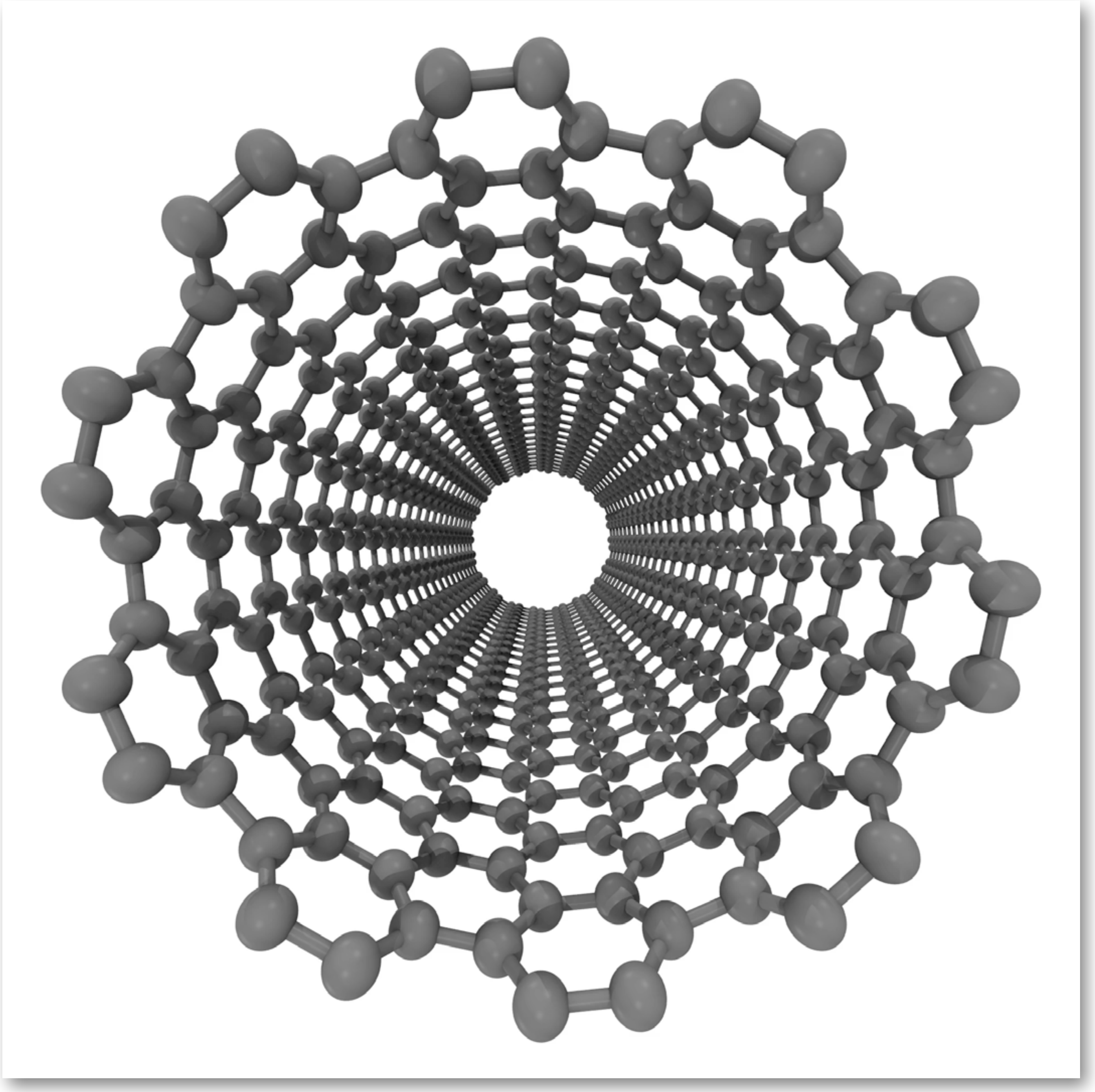
- Learning implies finding an algorithm (not necessarily machine learning) that is effective for the problem to be solved.

Molecular Dynamics Simulations



$$\frac{d\mathbf{x_p}}{dt} = \mathbf{u_p}$$

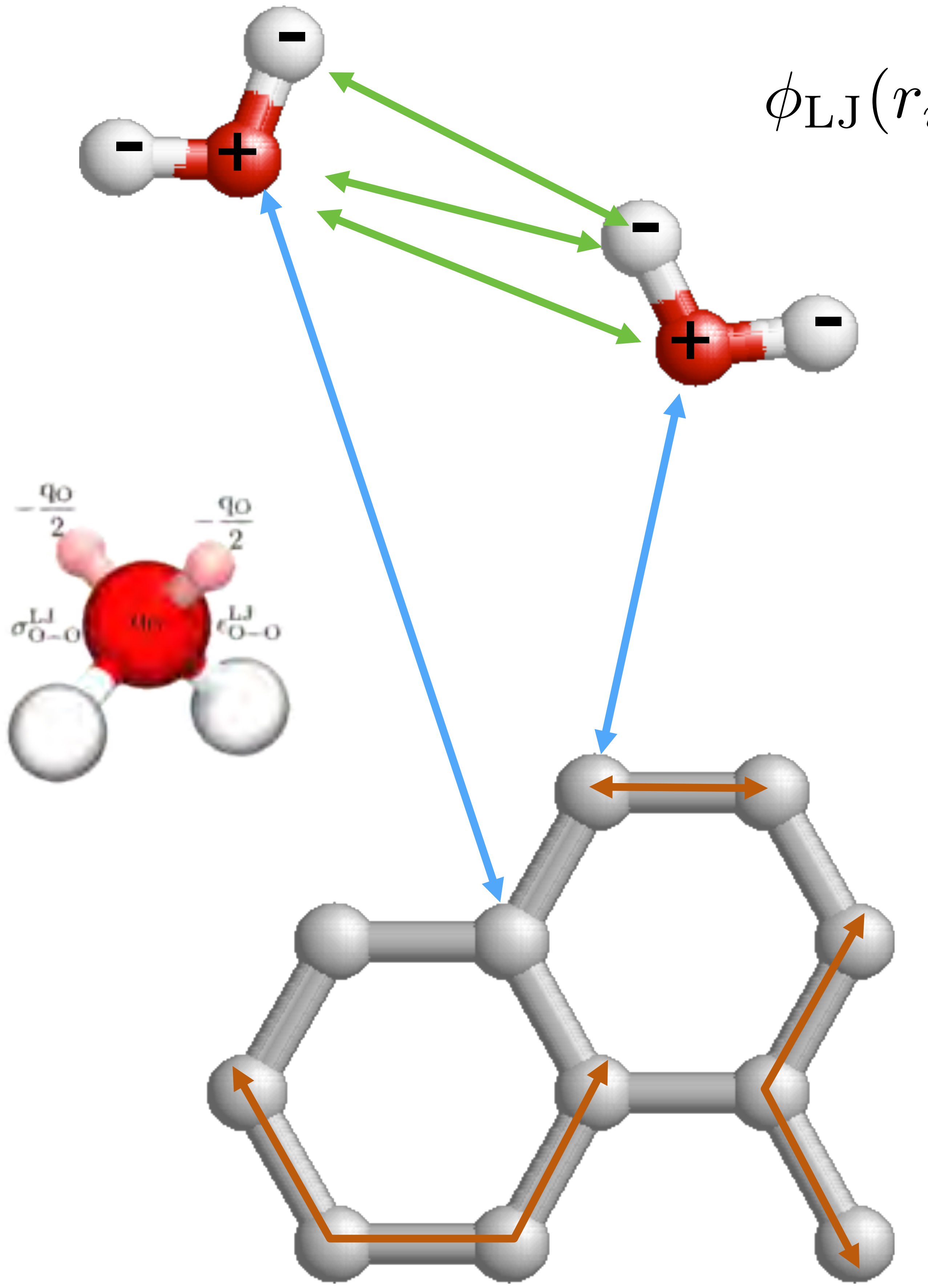
$$m\frac{d\mathbf{u_p}}{dt} = \sum_i^N \mathbf{F_p} \equiv -\frac{\partial U}{\partial \mathbf{x}_p}$$

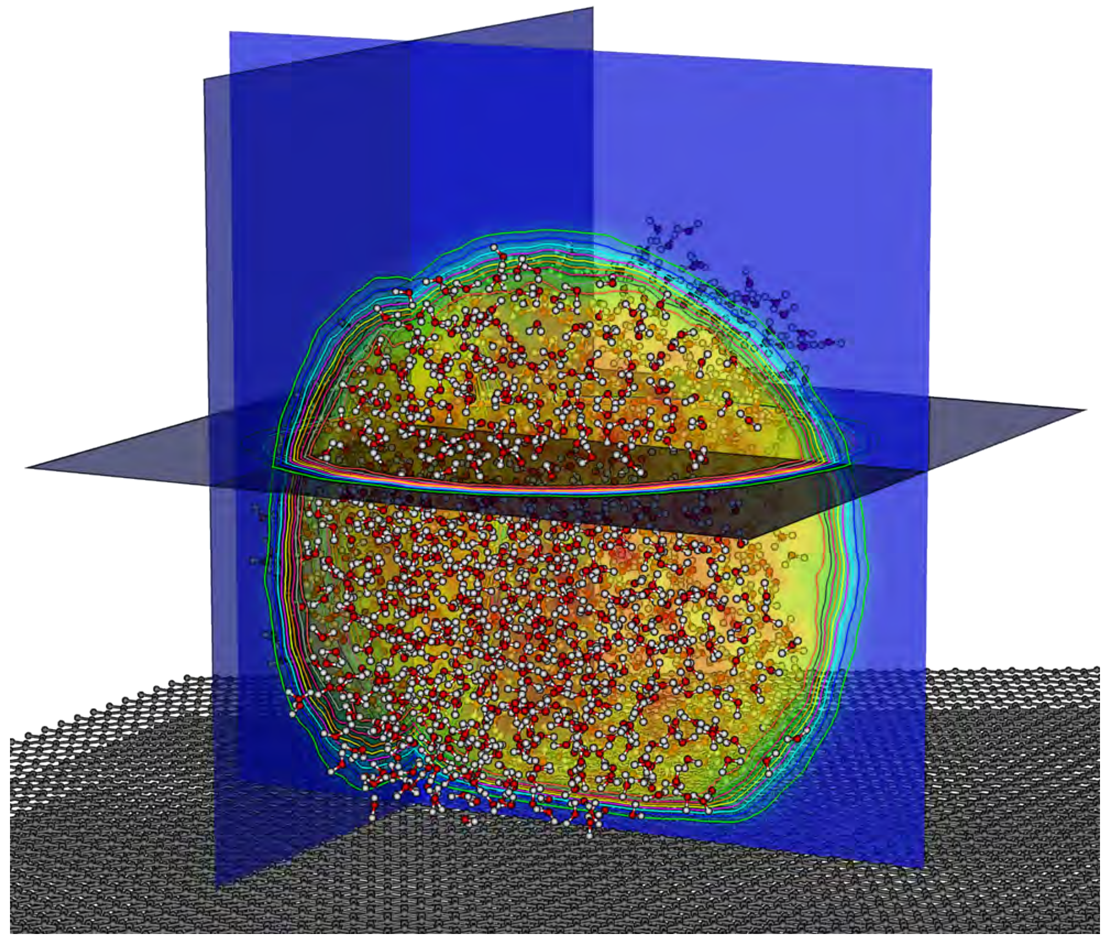


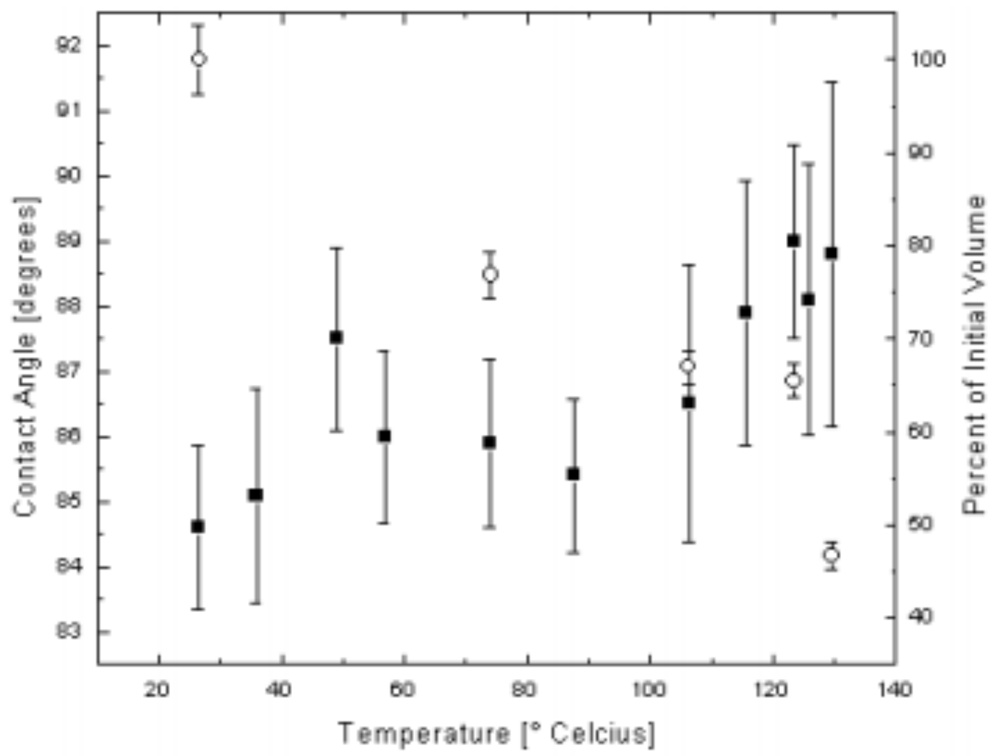
Parameters as Sources of Uncertainty (Water-Graphite Systems)

MODELLING **PARAMETRIC** **COMPUTATIONAL** **MEASUREMENT**

$$\phi_{\mathrm{LJ}}(r_{ij}) = 4\epsilon_{\mathrm{LJ}}\left[\left(\frac{\sigma_{\mathrm{LJ}}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{\mathrm{LJ}}}{r_{ij}}\right)^{6}\right]$$







K. Osborne III (2009)

Wetting Dependence on MD Potentials

NATURE | VOL 414 | 8 NOVEMBER 2001 | www.nature

Water conduction through the hydrophobic channel of a carbon nanotube

G. Hummer*, J. C. Rasaiah*† & J. P. Noworyta†

NANO LETTERS 2001 Vol. 1, No. 12 697−702

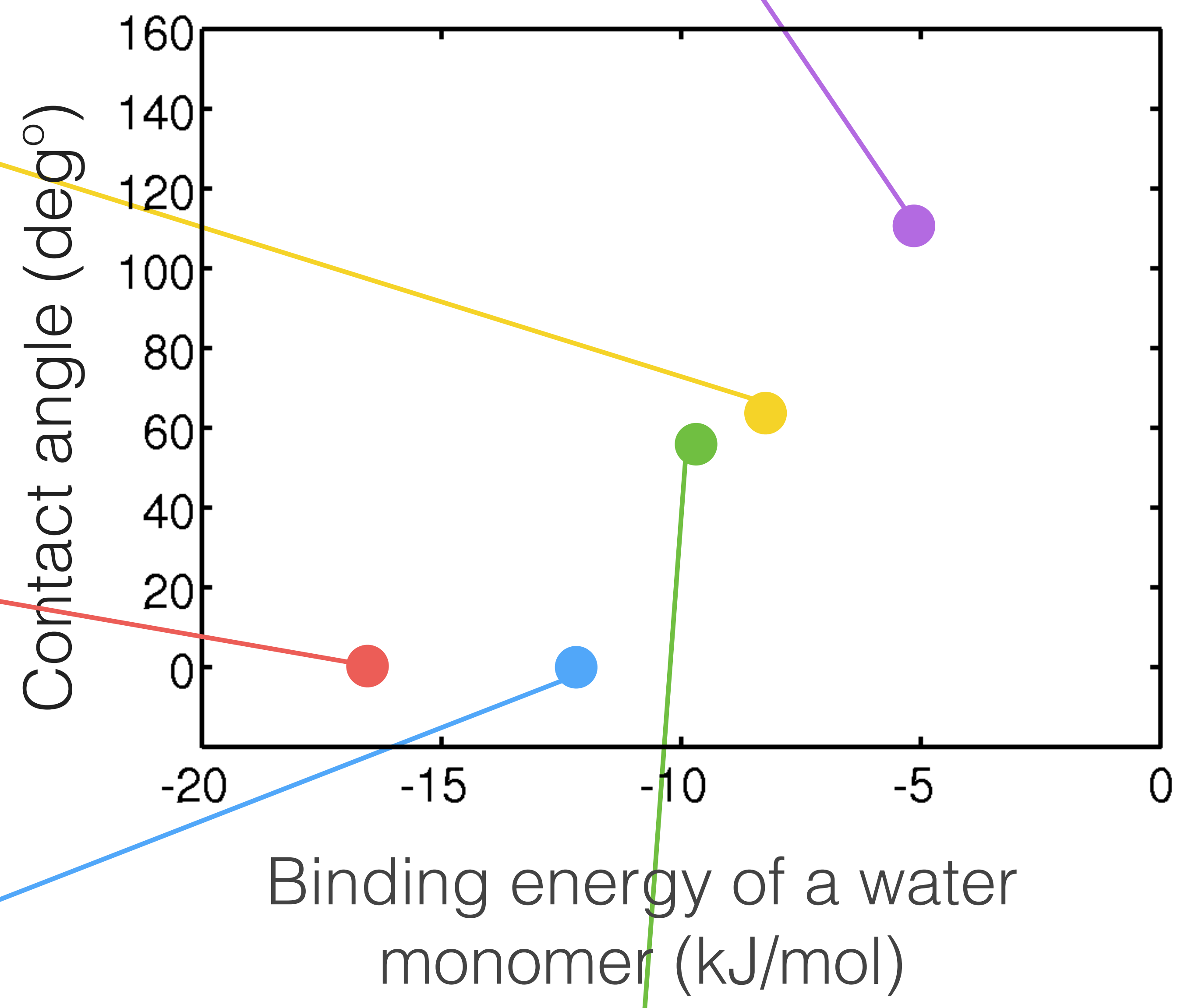
Molecular Dynamics Simulation of Contact Angles of Water Droplets in Carbon Nanotubes

Thomas Werder,*,† Jens H. Walther,† Richard L. Jaffe,‡ Timur Halicioglu,§ Flavio Noca,|| and Petros Koumoutsakos†,⊥

ELSEVIER 8 April 2002 Chemical Physics Letters 355 (2002) 445–448 CHEMICAL PHYSICS LETTERS www.elsevier.com/locate/cplett

Helical ice-sheets inside carbon nanotubes in the physiological condition

William H. Noon [a], Kevin D. Ausman [b], Richard E. Smalley [b], Jianpeng Ma [a,c,d,*]



ELSEVIER Chemical Physics 247 (1999) 413–430 Chemical Physics www.elsevier.nl/locate/chemphys

Scattering of water from graphite: simulations and experiments

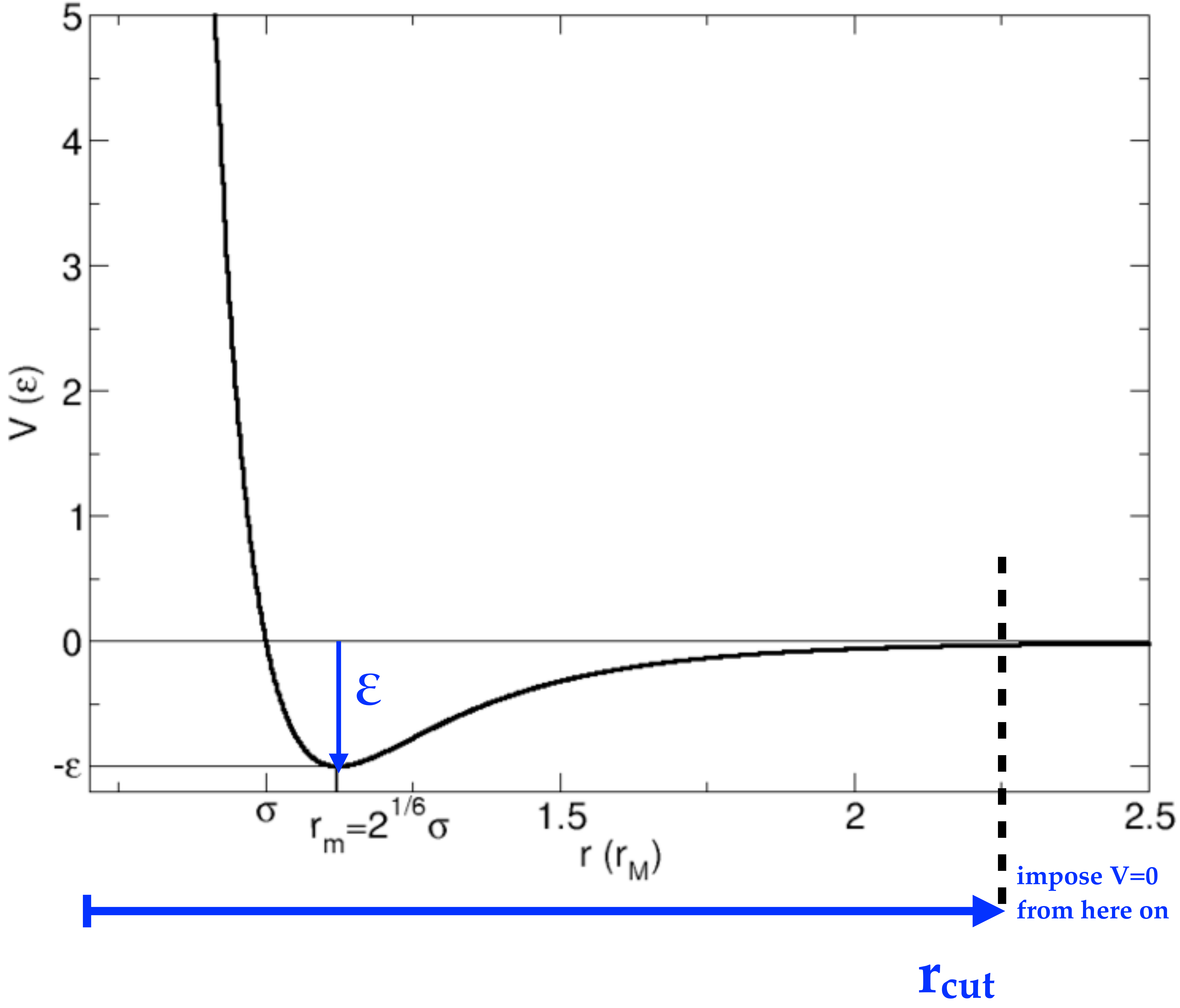
Nikola Marković *, Patrik U. Andersson, Mats B. Någård, Jan B.C. Pettersson [1]

ELSEVIER 27 October 2000 Chemical Physics Letters 329 (2000) 341–345 CHEMICAL PHYSICS LETTERS www.elsevier.nl/locate/cplett

Hydrogen bond structure of liquid water confined in nanotubes

M.C. Gordillo, J. Martí *

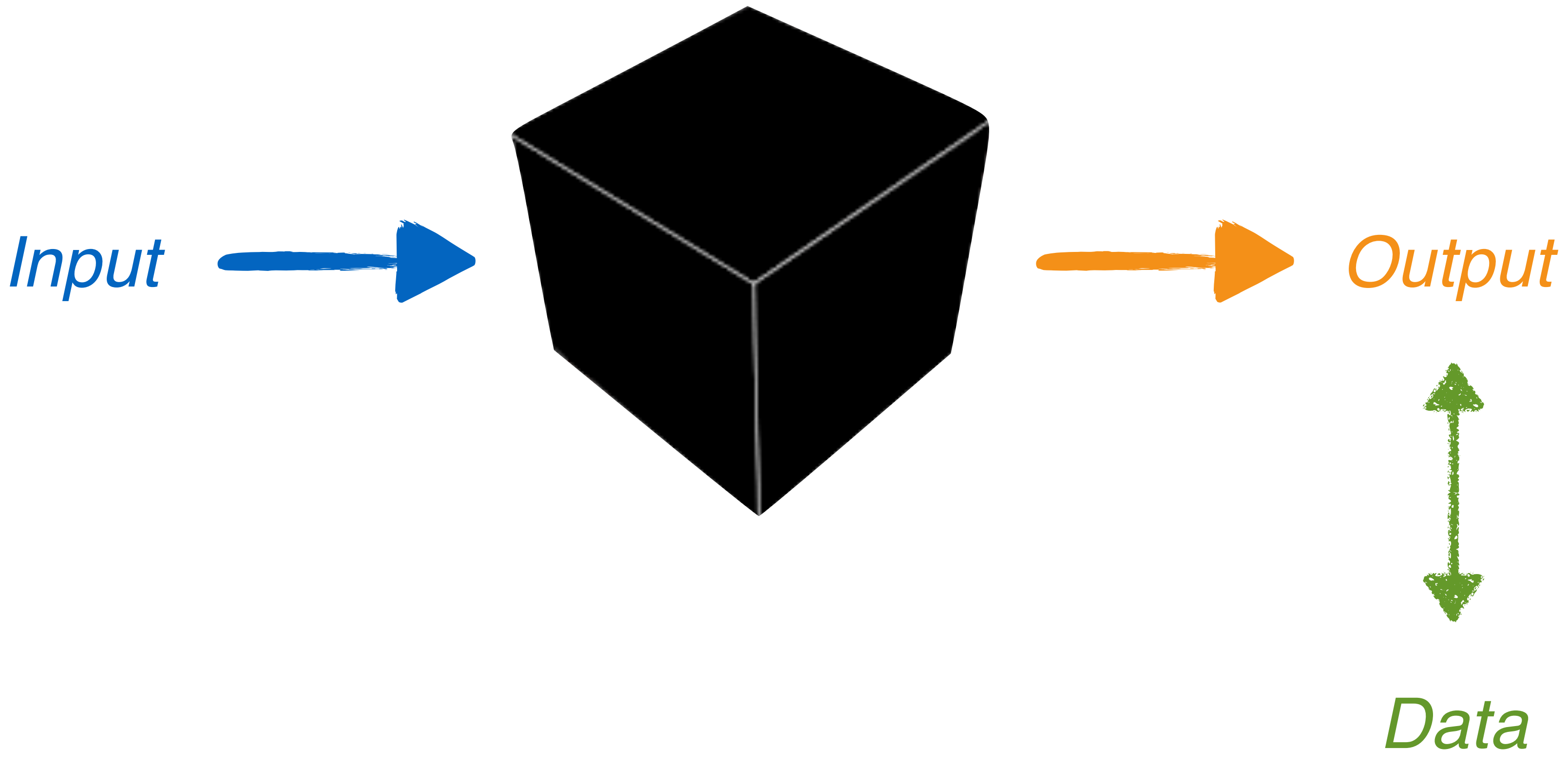
Lennard-Jones potential : well depth and cut-off



Bayesian Uncertainty Quantification: Calibration and model selection

Model with Parameters

Which parameters are “good” for the model?



compare with the data and decide

Result: a ***distribution*** over all possible parameters of the model

- **How to compare the output of the model with the data?**

Classical approach:

prediction error equation

$$D = M(\theta) + \varepsilon$$
$$\varepsilon \sim \mathcal{N}(0, \Sigma)$$

likelihood

$$p(D|\theta)$$

Bayes’ Theorem

posterior

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$$

Bayesian UQ: Calibration and Model Selection

Experimental Data: D

Use observations to select the model classes and estimate their parameter values such that the model predictions best fit the data

PARAMETER ESTIMATION

$$f\left(\theta_i | D, \mathcal{MD}_i\right) = \frac{f\left(D|\theta_i, \mathcal{MD}_i\right) \pi\left(\theta_i|\mathcal{MD}_i\right)}{f\left(D|\mathcal{MD}_i\right)}$$

Experiments

Physical limitations
Past studies
Expert elicitation

MODEL CLASS SELECTION

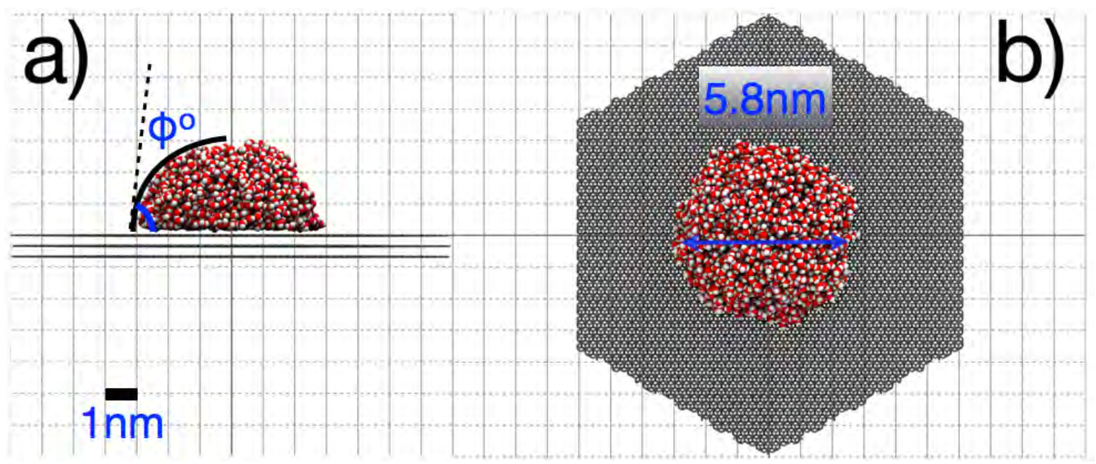
$$Pr\left(\mathcal{MD}_i|D\right) = \frac{f\left(D|\mathcal{MD}_i\right) Pr\left(\mathcal{MD}_i\right)}{f\left(D\right)}$$

Evidence of Model

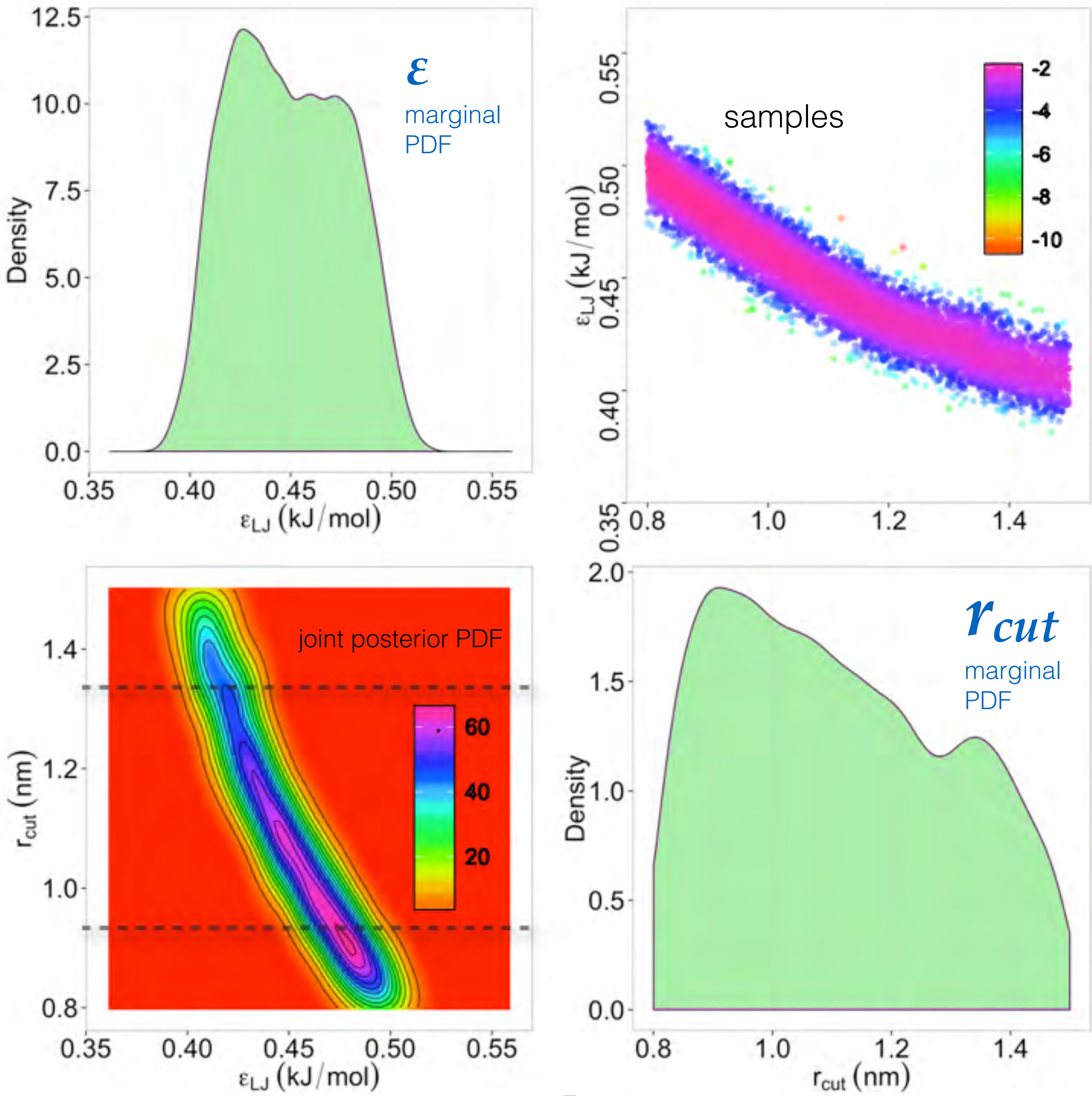
$$f\left(D|\mathcal{MD}_i\right) = \int f\left(D|\theta_i, \mathcal{MD}_i\right) \pi\left(\theta_i|\mathcal{MD}_i\right) d\theta_i$$

UQ

water contact angle

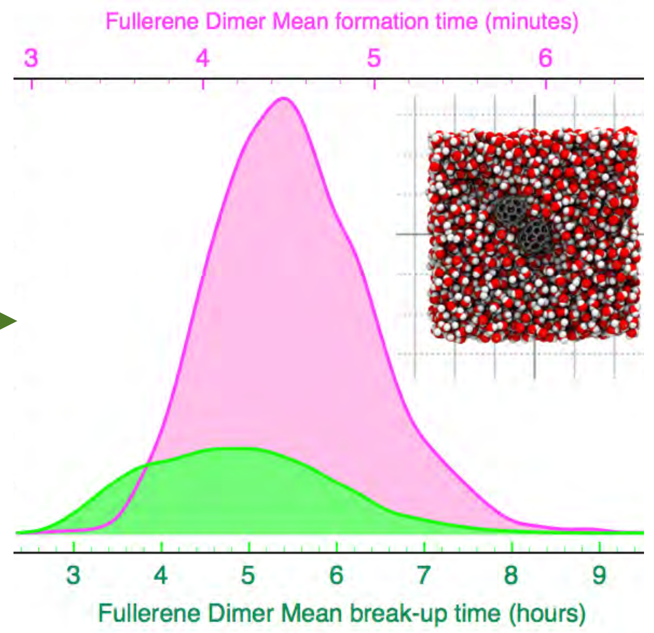


parameters: ε, r_{cut}



P

buckyball aggregation



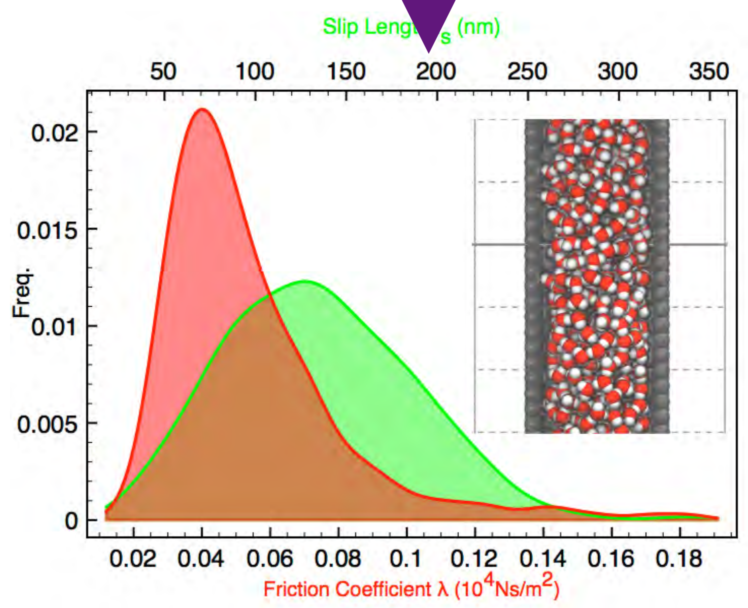
200 samples - 7 days

HPC challenges for UQ in nanoscale flows:

large, variable, computational cost per calibration or propagation sample

Resources: 1200 cores, 32GPUs

P



Friction coefficient and slip length of water inside CNTs

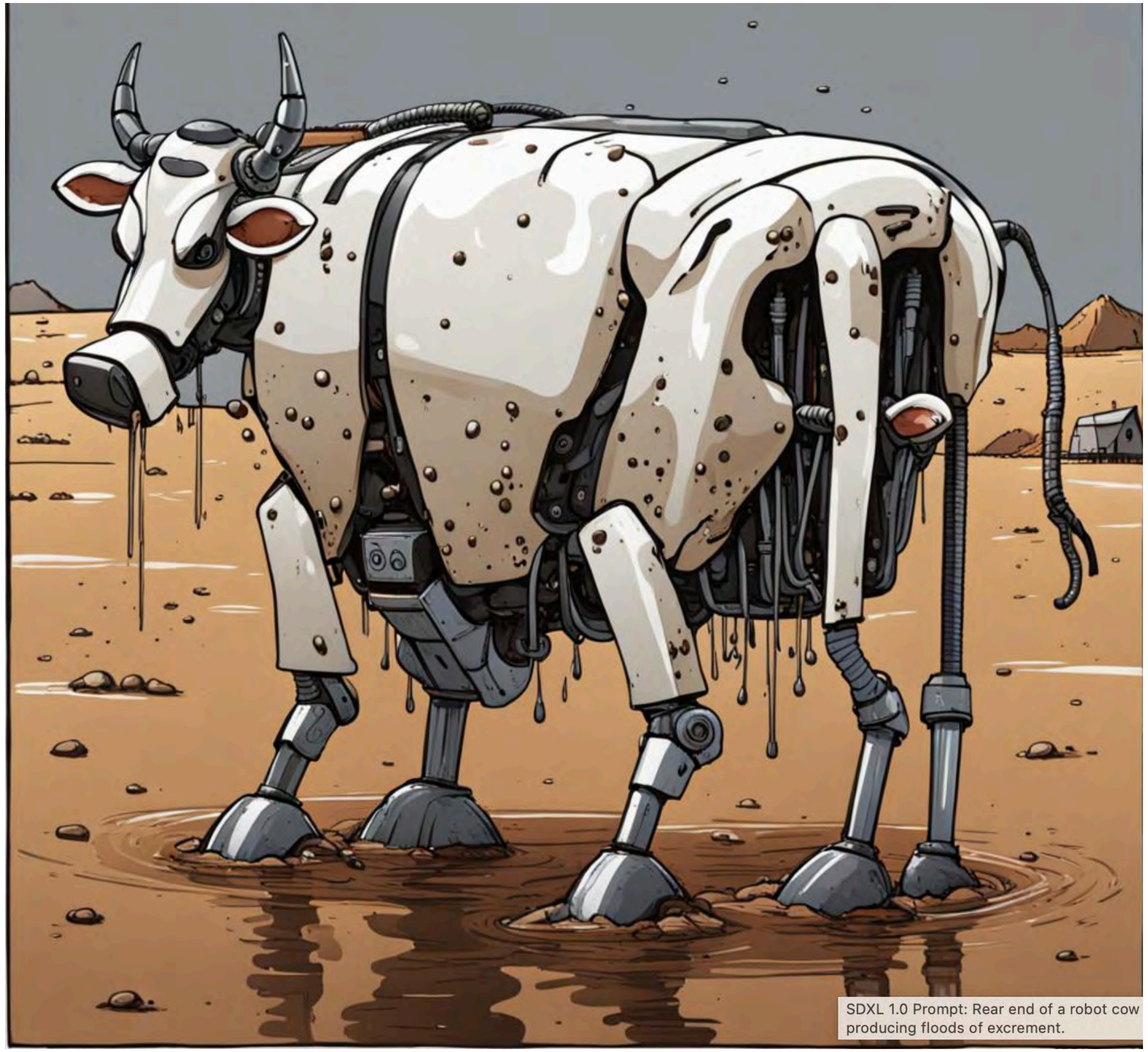
500 samples - 2 days

water transport in CNTs

Angelikopoulos, P., Papadimitriou, C. and Koumoutsakos, P. (2013), J. Phys. Chem. B

"Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell, "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" in Proceedings of the 2021 ACM conference on Fairness, Accountability, and Transparency (March 2021), 610-623."

Here's a metaphorical visualization inspired by the paper "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" The scene depicts a giant, digital parrot made up of code and algorithmic patterns, representing the vast scale and influence of large language models, as it oversees a landscape of intellectual debate and ethical considerations.



SDXL 1.0 Prompt: Rear end of a robot cow producing floods of excrement.

UNIVERSITY OF CAMBRIDGE | Study at Cambridge | About the University | Research at Cambridge | Quick links | Search

Home / Blogs / Alan Blackwell's blog

Department of Computer Science and Technology

Sign in with Raven

Oops! We Automated Bullshit.

are there any similarities between the multigrid cycle and unet?

ChatGPT

Yes, there are indeed similarities between the Multigrid Cycle and U-Net, particularly in their structural approaches to handling data across different scales or resolutions. Both methodologies are used in different contexts (Multigrid Cycle in numerical methods for solving partial differential equations, and U-Net in deep learning for biomedical image segmentation), but they share some conceptual underpinnings:

- **Hierarchical Processing**: Both techniques employ a hierarchical approach to process data. In the Multigrid Cycle, this involves solving the problem at various grid resolutions, progressively refining the solution from coarse to fine grids. U-Net also processes information at multiple resolutions, using a contracting path to capture context and a symmetric expanding path to enable precise localization.
- **Data Flow Across Scales**: Both methods facilitate the flow of information across different scales. The Multigrid Cycle explicitly solves the problem at multiple scales and uses interpolation and restriction operations to transfer information between coarse and fine grids. Similarly, U-Net architecture includes skip connections that directly concatenate feature maps from the contracting path to the corresponding layers in the expanding path, ensuring that fine-grained details are not lost during upsampling.
- **Efficiency in Handling Complex Structures**: The ability to process information at multiple scales makes both approaches particularly efficient in handling complex structures within the data. For instance, the Multigrid Cycle can efficiently solve partial differential equations with varying coefficients that may arise from complex geometries. U-Net, on the other hand, is effective in segmenting images with complex cellular structures where both local details and global context are important.
- **Goal of Improved Accuracy and Convergence**: Both methodologies aim to improve the accuracy and convergence of their respective tasks. The Multigrid Cycle seeks to accelerate the convergence of solutions to differential equations by leveraging solutions at multiple resolutions, while U-Net aims to improve segmentation accuracy by effectively combining local and global information through its architecture.

Despite these similarities, it's crucial to note the differences in their applications and specific methodologies. The Multigrid Cycle is a numerical algorithm designed for efficiently solving equations, while U-Net is a deep learning model architecture designed for image segmentation tasks. However, the conceptual overlap highlights how strategies for managing information across scales can be beneficial across different domains.


SCIENTIFIC COMPUTING
Mathematics
Exactness
Understanding
ALLOYS
ARTIFICIAL INTELLIGENCE
Architectures
Statistics
Goals

Thank you !